

**NASA - JSC  
TECHNICAL  
LIBRARY**

**Call No:**

TL  
797  
.B49  
1990  
v. 2  
pt. 2  
C. 1

# Beyond the Baseline

## *Proceedings of the Space Station Evolution Symposium*

Volume 2: Space Station Freedom  
Advanced Development Program

Part 2

**TECHNICAL LIBRARY  
BUILDING 45**

**FEB 21 1991**

**Johnson Space Center  
Houston, Texas 77058**

*Proceedings of a conference held at  
South Shore Harbour Resort  
and Conference Center  
League City, Texas  
February 6-8, 1990*

---

**NASA**

---

## Preface

This publication is a compilation of papers presented at the First Annual Space Station Evolution Symposium: Beyond the Baseline on February 6-8, 1990. The symposium focused on the presentation of results by the personnel responsible for advanced system studies and advanced development tasks within the Space Station Freedom Program. The symposium provided an opportunity for dialogue between the users, designers, and advanced planners for Station regarding the long-term utilization of Space Station Freedom.

The papers describe efforts included within the Level I Transition Definition Program to define and incorporate baseline design accommodations which satisfy the requirements associated with potential evolutionary paths, and to develop advanced technology which will enhance Space Station capabilities and enable its evolution. The papers describe work accomplished during fiscal year 1989 and were presented by those in Government, industry, and academia who performed the tasks.

This publication consists of two volumes. Volume 1 contains the results of the advanced system studies with the emphasis on reference evolution configurations, system design requirements and accommodations, and long-range technology projections. Volume 2 reports on advanced development tasks within the Transition Definition Program. Products of these tasks include: engineering fidelity demonstrations and evaluations on Station development testbeds and Shuttle-based flight experiments; detailed requirements and performance specifications which address advanced technology implementation issues; and mature applications and the tools required for the development, implementation, and support of advanced technology within the Space Station Freedom Program.

Dr. Earle K. Huckins III  
Director, Space Station Freedom Engineering  
Office of Space Flight  
NASA Headquarters

Listed below are the persons who made this symposium possible.

## COMMITTEE MEMBERS

### Conference Chair:

- Earle Huckins III  
*NASA Headquarters*

### Program Planning Committee Technical Chairs:

- Stephen Cook  
*NASA Headquarters*  
Evolution Systems Studies and  
Analysis
- Gregg Swietek  
*NASA Headquarters*  
Advanced Development  
Program

### Session Chairs:

- E. Brian Pritchard  
*NASA Langley Research  
Center*
- Barry Meredith  
*NASA Langley Research  
Center*
- Karen Brender  
*NASA Langley Research  
Center*
- Gregg Swietek  
*NASA Headquarters*

- Paul Neumann  
*Space Station Freedom  
Program Office*

- John Muratore  
*NASA Lyndon B. Johnson  
Space Center*

- Henry Lum  
*Ames Research Center*

- Wayne Zimmerman  
*NASA Headquarters*

- Dale Weathers  
*Space Station Freedom  
Program Office*

### Administrative Co-Chairs:

- Glenn Freedman  
*UH-Clear Lake*
- Carla Armstrong  
*Barrios Technology*

### Graphics Coordinator:

- Peter Colangelo  
*Omniplan Corporation*

## CONTENTS – Volume 2, Part 2

Title	Page
SPACE STATION FREEDOM ADVANCED DEVELOPMENT PROGRAM OVERVIEW .....	1
<b>SESSION IV: FLIGHT SYSTEMS AUTOMATION</b>	
<b>Session Chair: Mr. Paul Neumann, Space Station Freedom Program Office</b>	
DEVELOPMENT STATUS AUTOMATION ADVANCED DEVELOPMENT SPACE STATION FREEDOM ELECTRIC POWER SYSTEM .....	21
SPACE STATION FREEDOM ADVANCED DEVELOPMENT BIMONTHLY REPORT FOR SSM/PMAD AUTOMATION .....	37
AUTOMATION OF THE ENVIRONMENTAL CONTROL AND LIFE SUPPORT SYSTEM .....	59
PI-IN-A-BOX, AN EXPERT SYSTEM TO ADVISE ASTRONAUTS DURING EXPERIMENTS .....	79
REMOTE MANUPILATOR SYSTEM (RMS) AUTOMATION .....	121
THERMAL EXPERT SYSTEM - TEXSYS DEVELOPMENT AND TEST REVIEW .....	157
SUMMARY OF ASTRONAUTS' INPUTS CONCERNING AUTOMATION .....	173
<b>SESSION VI: GROUND OPERATIONS AUTOMATION</b>	
<b>Session Chair: Mr. John Muratore, NASA Lyndon B. Johnson Space Center</b>	
REAL TIME DATA SYSTEM .....	189
TRANSITION FLIGHT CONTROL ROOM AUTOMATION .....	221
INTELLIGENT COMPUTER-AIDED TRAINING .....	239
PLATFORM MANAGEMENT SYSTEM EVOLUTION .....	255
AUTOMATED PLATFORM MANAGEMENT SYSTEM SCHEDULING .....	299
CONCEPTS IN DISTRIBUTED SCHEDULING AND CONTROL .....	325

Title	Page
-------	------

**SESSION VIII: SPACE STATION INFORMATION SYSTEMS**

**Session Chair: Mr. Robert Nelson, Space Station Freedom Program Office**

DATA MANAGEMENT SYSTEM (DMS) ADVANCED OPERATING SYSTEM STUDY .....	343
OMS ADVANCED AUTOMATION: FDIR PROTOTYPING .....	363
OPERATIONS MANAGEMENT SYSTEM; EVENT EVALUATOR AND SCHEDULER .....	385
AUTOMATED SOFTWARE DEVELOPMENT WORKSTATION (ASDW) .....	405
EVOLUTION PATHS FOR ADVANCED AUTOMATION .....	421

**SESSION IX: ADVANCED AUTOMATION ENVIRONMENTS**

**Session Chair: Dr. Henry Lum, NASA Ames Research Center**

ART/Ada and CLIPS/Ada .....	437
KNOWLEDGED-BASED SYSTEMS VERIFICATION AND VALIDATION .....	445
THE SOFTWARE SUPPORT ENVIRONMENT DESIGN KNOWLEDGE CAPTURE (SSE/DKC) PROJECT .....	461
ADVANCED ARCHITECTURE TESTBED .....	491
SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS .....	515
EOS PROCESSOR EXPERIMENT (ISES) .....	527
AUTONOMOUS CONTROL (FORMERLY LAUNCH OPERATIONS) .....	565

**SESSION X: TELEROBOTICS TECHNOLOGY AND APPLICATIONS**

**Session Chair: Mr. Wayne Zimmerman, Office of Space Station**

TELEROBOTIC SYSTEM TECHNOLOGY .....	587
ROBOTIC ASSEMBLY OF LARGE SPACE STRUCTURES .....	651

Time	Topic	Presenter
<b>Monday February 5, 1990</b>		
6:00 p.m. - 9:00 p.m.	Registration	
<b>Tuesday February 6, 1990</b>		
7:30 a.m.	Registration	
8:30	OPENING SESSION	
	Welcoming Remarks	Dr. Aaron Cohen <i>Director, NASA Johnson Space Center</i>
9:00	Keynote Address	Dr. William B. Lenoir <i>Associate Administrator for Space Flight and Acting Associate Administrator for Space Station</i>
9:30	Space Station Freedom Program Overview	Mr. Richard H. Kohrs <i>Director, Space Station Freedom</i>
10:00	Break	
10:30	Human Exploration Mission Planning	Dr. Franklin D. Martin <i>Associate Administrator for Exploration</i>
11:00	Mission to Planet Earth	Dr. Shelby G. Tilford <i>Manager, Geostationary Observations, Mission to Planet Earth</i>
11:30	Space Station Freedom Evolution	Dr. Earle K. Huckins III <i>Director, Strategic Plans and Programs Division Office of Space Station, NASA Headquarters</i>
12:00 - 1:30	Lunch - Harbour Club	
1:30 - 4:30	SESSION I — EVOLUTION MISSION AND PLANNING Session Chair: Mr. E. Brian Pritchard <i>NASA Langley Research Center</i>	
	Long-range Planning for Science Utilization on Space Station	Mr. Robert C. Rhome <i>NASA Headquarters</i>
	Space Station Accommodation of Lunar/Mars Exploration Program	Mr. Lewis Peach <i>NASA Headquarters</i>
	Advanced Transporation Systems	Mr. Darrell R. Branscome <i>NASA Headquarters</i>
	Future European Manned Space Infrastructure	Mr. Jacques Collet <i>European Space Agency</i>
	Break	
	Japan's Future Space Activities	Mr. Masatoshi Saito <i>National Space Development Agency of Japan</i>
	Canadian Space Activities in the 21st Century	Mr. R. Brian Erb <i>Canadian Liaison Office</i>
	Space Station as a Technology Testbed	Dr. Judith Ambrus <i>NASA Headquarters</i>
	Life Sciences Planning for Manned Missions	Dr. Arnauld E. Nicogossian <i>NASA Headquarters</i>
1:30 - 4:30	SESSION II — TECHNOLOGY AND ADVANCED DEVELOPMENT OVERVIEW Session Chair: Mr. Gregg Swietek <i>NASA Headquarters</i>	
	Office of Aeronautics and Space Technology (OAST) Programs	Dr. Judith Ambrus <i>NASA Headquarters</i>
	OAST Systems Autonomy and Telerobotics Programs	Dr. Mel Montemerlo <i>NASA Headquarters</i>
	Office of Space Flight Advanced Development Activities	Ms. Pat Connor <i>NASA Headquarters</i>
	Space Station Freedom (SSF) Flight Telerobotic Servicer	Dr. Harry McCain <i>NASA Goddard Space Flight Center</i>
	SSF Advanced Development Program Overview	Mr. Gregg Swietek <i>NASA Headquarters</i>
5:30	Reception	
6:30	Banquet with Speaker	

Wednesday February 7, 1990

8:30 - 12:00

SESSION III — CONFIGURATION EVOLUTION

Session Chair: Mr. E. Brian Pritchard  
NASA Langley Research Center

Space Station Freedom Integrated Research and Development Growth	Mr. Rudy Saucillo McDonnell Douglas, Washington, D.C.
Space Station Transportation Node Concepts and Analysis	Mr. William Cirillo NASA Langley Research Center
Servicing Capability for the Evolutionary Space Station	Mr. Ted Grems McDonnell Douglas Greenbelt, MD
Evolutionary Space Station Fluids Management	Mr. Steve Stevenson NASA Lewis Research Center
Space Station Logistics Systems Evolution	Mr. Michael Tucker NASA Marshall Space Flight Center
Space Transfer Vehicle Accommodations at Transportation Nodes	Mr. Uwe Hueter NASA Marshall Space Flight Center
A Radiological Assessment of Space Nuclear Power Operations Near Space Station	Mr. Steve Stevenson NASA Lewis Research Center
Platform Evolution Studies	Ms. Barbara Walton NASA Goddard Space Flight Center

8:30 - 12:00

SESSION IV — FLIGHT SYSTEMS AUTOMATION

Session Chair: Mr. Paul Neumann  
Space Station Freedom Program Office

Autonomous Power Management and Distribution (PMAD)	Mr. Jim Dolce / Mr. Gale Sundberg / Mr. Jim Kish NASA Lewis Research Center
Laboratory/Habitation Module PMAD Automation	Mr. Bryan Walls NASA Marshall Space Flight Center
Environmental Control and Life Support System (ECLSS)	Mr. Brandon Dewberry NASA Marshall Space Flight Center
PI-in-a-Box	Dr. Larry Young / Dr. Silvano Colombano Massachusetts Institute of Technology / NASA Ames Research Center
RCS/RMS Automation using Procedural Reasoning	Mr. H. K. Hiers NASA Johnson Space Center
Thermal Control Expert System	Dr. John Bull / Ms. Kathy Healey / Mr. Jeff Dominick NASA Ames Research Center / NASA Johnson Space Center
Summary of Astronauts' Inputs Concerning Automation	Mr. Dave Weeks NASA Marshall Space Flight Center

12:00 - 1:00

Lunch - Harbour Club

1:30 - 4:30

SESSION V — SYSTEM EVOLUTION

Session Chair: Mr. Barry D. Meredith  
NASA Langley Research Center

Assuring Data Transparency Through Design Methodologies	Mr. Allen Williams Harris Corporation
EVA Systems	Mr. Michael Rouen NASA Johnson Space Center
Data Management Systems	Ms. Katherine Douglas NASA Johnson Space Center
Active Thermal System	Mr. Richard L. Bullock NASA Johnson Space Center
Break	
Guidance Navigation and Control	Mr. Jerry Kennedy TRW, Inc.
Communications and Tracking	Mr. William Culpepper NASA Johnson Space Center
Structural Analysis of Evolution Station Concepts	Mr. Paul Cooper NASA Langley Research Center
Environmental Control and Life Support System	Mr. Paul Wieland NASA Marshall Space Flight Center

Time	Topic	Presenter
<b>Wednesday February 7, 1990 (continued)</b>		
1:00 - 4:30	<b>SESSION VI — GROUND OPERATIONS AUTOMATION</b> Session Chair: Mr. John Muratore <i>NASA Johnson Space Center</i>	
	Real Time Data Systems for Mission Control	Mr. John Muratore / Mr. Troy Heindel <i>NASA Johnson Space Center</i>
	Transition Flight Control Room Automation	Mr. Al Brewer <i>NASA Johnson Space Center</i>
	Intelligent Computer-Aided Training Environment	Mr. Bob Savely <i>NASA Johnson Space Center</i>
	Platform Management System (PMS) Evolution	Mr. John Hartley / Mr. Mike Tilley <i>NASA Goddard Space Flight Center</i>
	Automated PMS Scheduler	Dr. Larry Hull <i>NASA Goddard Space Flight Center</i>
	Concepts in Distributed Planning and Control	Dr. Elaine Hansen / Dr. Larry Hull <i>NASA Goddard Space Flight Center</i>
<b>Thursday February 8, 1990</b>		
8:30 - 11:00	<b>SESSION VII — OPERATIONS EVOLUTION</b> Session Chair: Ms. Karen Brender <i>NASA Langley Research Center</i>	
	Operations Analysis for Evolution Station Concepts	Ms. Karen Brender <i>NASA Langley Research Center</i>
	Vehicle Processing Operations Database (VPOD)	Mr. George Ganoe <i>NASA Langley Research Center</i>
	On-orbit Assembly and Servicing Task Definition	Mr. Rick Vargo <i>McDonnell Douglas, Kennedy Space Center</i>
	Advanced Robotics for In-Space Vehicle Processing	Dr. Jeffrey Smith <i>NASA Jet Propulsion Laboratory</i>
	Advanced Automation for In-Space Vehicle Processing	Dr. Michael Sklar <i>McDonnell Douglas, Kennedy Space Center</i>
	Space Vehicle Deployment from Space Station	Mr. Paul Henry <i>NASA Jet Propulsion Laboratory</i>
	Graphical Analysis of Mars Vehicle Assembly	Mr. Kevin Lewis <i>NASA Johnson Space Center</i>
8:30 - 11:00	<b>SESSION VIII — SPACE STATION INFORMATION SYSTEMS</b> Session Chair: Mr. Del Weathers <i>Space Station Freedom Program Office</i>	
	Data Management System Advanced Automation	Ms. Katherine Douglas / Mr. Terry Humphrey <i>NASA Johnson Space Center</i>
	Operations Management System (OMS) Global Fault Detection / Isolation	Mr. Matt Hanson
	OMS Event Evaluator and Scheduler	Mr. Rick Eckelkamp <i>NASA Johnson Space Center</i>
	Automated Software Development Workstation	Mr. Ernie Fridge <i>NASA Johnson Space Center</i>
	Technical and Management Information System Design Knowledge Capture	Dr. John Boose / Dr. Jeff Bradshaw / Dr. David Sheema / Dr. Stanley Covington <i>Boeing Advanced Technology Center</i>
	Evolution Paths for Advanced Automation	Ms. Kathy Healey <i>NASA Johnson Space Center</i>
11:00 - 12:00	Lunch - Harbour Club	
12:00 - 4:00	<b>SESSION IX — ADVANCED AUTOMATION ENVIRONMENTS</b> Session Chair: Dr. Henry Lum <i>Ames Research Center</i>	
	CLIPS/Ada Programming Tool	Mr. Chris Culbert <i>NASA Johnson Space Center</i>
	ART/Ada Programming Tool	Mr. Chris Culbert <i>NASA Johnson Space Center</i>

Time	Topic	Presenter
<b>Thursday February 8, 1990</b> <i>(continued)</i>		
	Knowledge-Based System Verification and Validation	Ms. Sally Johnson <i>NASA Langley Research Center</i>
	Intelligent Systems Engineering Methodology	Dr. Bruce Bullock <i>ISX, Inc.</i>
	Software Support Environment Design Knowledge Capture	Mr. Tom Dollman <i>NASA Marshall Space Flight Center</i>
	Advanced DMS Architectures Testbed	Mr. Terry Grant <i>NASA Ames Research Center</i>
	Spaceborne Autonomous Multiprocessor System	Mr. Alan Fernquist <i>NASA Ames Research Center</i>
	Information Sciences Experiment System Architecture	Mr. Nick Murray / Mr. Steve Katzberg <i>NASA Langley Research Center</i>
12:00 - 3:00 <b>SESSION X — TELEROBOTICS TECHNOLOGY AND APPLICATIONS</b>		
	Session Chair: Mr. Wayne Zimmerman <i>NASA Headquarters</i>	
	JPL Shared Control Architecture	Dr. Paul Backes / Dr. Samad Hayati <i>NASA Jet Propulsion Laboratory</i>
	JPL/KSC Robotic Inspection	Mr. Brian Wilcox / Mr. Leon Davis <i>NASA Jet Propulsion Laboratory / NASA Kennedy Space Center</i>
	Robotic Assembly of Large Space Structures	Mr. Ralph Will / Mr. Marvin Rhodes <i>NASA Langley Research Center</i>
	Space Station IVA Payload Robot	Mr. E. C. Smith <i>NASA Marshall Space Flight Center</i>
	Advanced Human-System Interface	Dr. Mike McGreevey <i>NASA Ames Research Center</i>

**SESSION VIII**  
**SPACE STATION INFORMATION SYSTEMS**

**Session Chair:**  
**Mr. Robert Nelson**  
**Space Station Freedom Program Office**

# **DATA MANAGEMENT SYSTEM (DMS)**

## **ADVANCED OPERATING SYSTEM STUDY**

**Terry D. Humphrey  
NASA Johnson Space Center  
Spacecraft Software Division  
10/31/89**

DATA MANAGEMENT SYSTEM (DMS)  
ADVANCED OPERATING SYSTEM STUDY

ABSTRACT

The Space Station Freedom's Data Management System (DMS) provides the common hardware and software needed by onboard systems such as the guidance, navigation and flight control system and the environmental control and life support system. One of the most important of the common software items used by all onboard systems is the DMS standard data processor's operating system. A future upgrade of the DMS will be needed to accommodate significantly more onboard application software. The most cost-effective approach is to insert additional processors in existing onboard computers using the Intel Multibus II backplane. This will require an upgrade of the operating system to include multiple processor management. In order to minimize changes to the existing application software and other DMS system software, this operating system upgrade must consider the combination of all the old requirements with the new. The purpose of this study is to investigate the current and future operating system requirements for the space station DMS and to identify viable solutions to the problems of acquiring, transitioning to, and maintaining an advanced operating system which meets these requirements.

# INTRODUCTION

- FUTURE UPGRADE OF DMS NEEDED TO ACCOMMODATE SIGNIFICANTLY MORE ONBOARD SOFTWARE.
- BASICALLY TWO SOLUTIONS
  - #1 ADD MORE SINGLE-PROCESSOR COMPUTERS TO NETWORK
  - #2 INSERT ADDITIONAL PROCESSOR CARDS USING MULTIBUS II
- SOLUTION #2 COSTS LESS AND REQUIRES LESS ELEC. POWER
- ADDING MULTIPLE PROCESSOR MANAGEMENT TO OS IS SIGNIFICANT AND MUST NOT DISTURB ANY OLD REQUIREMENTS
- PURPOSE OF THIS STUDY
  - TO IDENTIFY CURRENT AND FUTURE OS REQUIREMENTS
  - TO IDENTIFY VIABLE SOLUTIONS TO ACQUIRING, TRANSITIONING TO, AND MAINTAINING AN ADVANCED OS

# NOTES - INTRODUCTION

To accommodate significantly more software onboard the space station in the future, the DMS will need to be upgraded. There are basically two solutions. The first is to add more of the single-processor computers to the global network. The second is to insert additional processor cards in the existing onboard computers using the Multibus II backplane and to upgrade the DMS Operating System to include multiple processor management functions.

The second solution costs less and requires less electrical power because each time a new single-processor computer is added to the network it requires another network interface unit. It is very important to note that adding multiple processor management is a significant change and must not disturb any other existing requirements on the operating system in order to minimize changes to existing application software. Therefore the purpose of this study is to identify current and future operating system requirements and to identify viable solutions to acquiring, transitioning to, and maintaining an advanced operating system that satisfies the combination of these requirements.

# **APPROACH**

- **IDENTIFY EXISTING AND FUTURE OPERATING SYSTEM REQUIREMENTS**
- **IDENTIFY CANDIDATE DESIGNS TO MEET REQUIREMENTS**
- **EVALUATE COMMERCIAL CANDIDATES USING VENDOR LITERATURE AND OUR OPERATING SYSTEM REQUIREMENTS**
- **IDENTIFY AND EVALUATE ACQUISITION AND MAINTENANCE APPROACHES**
- **WRITE FINAL REPORT**

## NOTES - APPROACH

The first step of this task was to identify existing and future operating system requirements. Second, we identified candidate designs to meet those requirements. Thirdly, we evaluated commercial candidate operating systems using vendor literature and our identified requirements. Fourthly, we identified and evaluated acquisition and maintenance approaches. And finally we wrote a report.

# **MAJOR HIGH-LEVEL REQUIREMENTS FOR "INITIAL DMS OS"**

- **POSIX-COMPLIANT**
- **ADA LANGUAGE SUPPORT, ESPECIALLY ADA MULTI-TASKING**
- **MULTI-PROGRAMMING**
- **REAL-TIME PROCESS-CONTROL APPLICATION SUPPORT**
- **HOSTED ON INTEL 80386 PROCESSOR**
- **NO VIRTUAL MEMORY PAGING/SWAPPING TO MASS STORAGE**

# - NOTES - MAJOR HIGH-LEVEL REQUIREMENTSFOR "INITIAL DMS OS"

The current high-level requirements for the initial operating system are that it shall:

- (1) be POSIX-compliant, i.e., it shall satisfy the current standards and be upgradeable to future POSIX standards.
- (2) support the Ada language requirements, e.g., Ada multi-tasking via threads or light-weight processes.
- (3) support multiple Ada programs running on a single processor computer.
- (4) support multiple real-time process-control application programs by providing cyclic execution with minimum jitter in processing as well as in input and output transactions.
- (5) run on an Intel 80386 processor and use the 32-bit protected mode.
- (6) and not perform virtual memory paging/swapping to mass storage.

# **MAJOR PROBLEMS OF THE "INITIAL DMS OPERATING SYSTEM"**

- **POSIX AND ADA ARE FAIRLY NEW TECHNOLOGIES**
- **THE COMBINATION OF ALL THESE MAJOR REQUIREMENTS INTO AN OPERATING SYSTEM FOR THE INTEL 80386 HAS NOT BEEN DONE AND WILL BE DIFFICULT BECAUSE:**
  - **ADA LANGUAGE DOES NOT SUPPORT WELL (ACCURATELY AND EFFICIENTLY) THE CYCLIC EXECUTION OF REAL-TIME PROCESS-CONTROL TASKS**
  - **ADA LANGUAGE DOES NOT SUPPORT MULTI-PROGRAMMING**
  - **POSIX CURRENTLY DOES NOT SUPPORT REAL-TIME PROCESS-CONTROL APPLICATIONS**

# - NOTES - MAJOR PROBLEMS FOR THE"INITIAL DMS OPERATING SYSTEM"

Some major problems exist in meeting these requirements for the initial operating system. One problem is that POSIX and Ada are fairly new technologies and there are not any mature implementations yet.

Another major problem is that the combination of all these standards and requirements into an operating system for the Intel 80386 has not been done yet by any vendor and will be difficult because :

(1) The Ada language does not support well the cyclic execution of real-time process-control tasks due to problems with the accuracy of the Ada delay statement.

(2) The Ada language does not support multi-programming.

And (3) the POSIX standard does not currently support real-time process-control application requirements, although an IEEE POSIX real-time subgroup is working on this.

# **MAJOR HIGH-LEVEL ADDITIONAL REQUIREMENTS FOR THE "ADVANCED DMS OPERATING SYSTEM"**

- **MULTIPLE PROCESSOR SUPPORT**
  - **MANAGES MULTIPLE APPLICATION PROCESSORS PER FLIGHT COMPUTER**
  - **MANAGES EXECUTION OF ONE APPLICATION PROGRAM PER PROCESSOR**
  - **MANAGES INPUT/OUTPUT UNITS SHARED BY MULTIPLE APPLICATION PROCESSORS WITHIN A COMPUTER, SUCH AS:**
    - **GLOBAL NETWORK I/O UNIT**
    - **LOCAL BUS I/O UNIT(S)**

**- NOTES -**

**MAJOR HIGH-LEVEL ADDITIONAL  
REQUIREMENTS FOR THE  
"ADVANCED DMS OPERATING SYSTEM"**

A major high-level requirement for the Advanced DMS Operating System is to support multiple processors. This means it must manage multiple application processors per flight computer. To reduce risks, the first multiple processor capability should be limited to management of one application program per processor. And should manage the sharing of the following input/output units among each application program/application processor: the global network I/O unit and the local bus I/O unit(s).

# **MAJOR PROBLEMS FOR THE "ADVANCED DMS OPERATING SYSTEM"**

- **SHARES SAME PROBLEMS AS "INITIAL DMS OPERATING SYSTEM"  
UNLESS SOLUTIONS ARE FOUND.**
- **POSIX DOES NOT CURRENTLY SUPPORT MANAGEMENT OF  
MULTIPLE APPLICATION PROCESSORS PER COMPUTER AND  
SHARED I/O UNITS.**

# - NOTES - MAJOR PROBLEMS FOR THE "ADVANCED DMS OPERATING SYSTEM"

The major problems facing the Advanced DMS Operating System will be the same as those for the initial one unless solutions are found. Another problem is that POSIX does not currently support the management of multiple application processors per computer and input/output units shared by them.

# ACQUISITION AND MAINTENANCE

- CRITICALITY OF OS MAKES TESTING AND CONFIGURATION CONTROL PARAMOUNT AMONG ISSUES RELATED TO ACQUIRING AND MAINTAINING THE OS.
  - AT LEAST, THE SOURCE CODE MUST BE EXAMINED AND PLACED UNDER CONFIGURATION CONTROL BY NASA CONTRACTOR.
- ALSO OF CONCERN IS ABILITY TO CORRECT PROBLEMS QUICKLY.
- TWO POSSIBLE SOLUTIONS:
  - OBTAIN OS WHICH MEETS REQUIREMENTS, AND THEN AT LEAST HAVE THE OPTION OF MAINTAINING IT WHEN NEEDED. THE DMS CONTRACTOR WOULD NEED TO HAVE THE PERSONNEL AND EXPERTISE TO CHANGE THE OS AT WILL.
  - OBTAIN OS WHICH MEETS MANY REQUIREMENTS, AND THEN MODIFY IT TO MEET THE REST. IT WOULD BE MAINTAINED BY THE DMS CONTRACTOR.

# - NOTES - ACQUISITION AND MAINTENANCE

The criticality of the DMS Operating System makes testing and configuration control paramount among the issues related to acquiring and maintaining the operating system. Therefore, at the least, the source code must be examined and placed under configuration control by the NASA contractor.

Also of concern is the ability to quickly correct problems that are found in the operating system. There are at least two possible solutions. The first is to obtain an operating system which meets the requirements, and then at least have the option of maintaining it when needed. The DMS contractor would need to have the personnel and expertise to change the operating system if required. The second solution is to obtain an operating system which meets many of the requirements and then modify it to meet the rest. In this case, the operating system would be maintained by the DMS contractor.

# CONCLUSIONS

- OF THE COMMERCIAL OPERATING SYSTEMS WE INVESTIGATED VIA VENDOR LITERATURE, THE FOLLOWING CAME CLOSEST TO MEETING OUR SET OF REQUIREMENTS:

REGULUS, GALAXY, VxWORKS, HARRIS, DDC-I, and LynxOS

- THE FOLLOWING COMMERCIAL OPERATING SYSTEMS WERE DETERMINED TO BE UNSUITABLE:
  - DEC VAXELN: ONLY AVAILABLE ON VAX
  - CMU MACH: UNIX KERNEL ONLY SLIGHTLY MODIFIED, PERFORMANCE-HARMING FEATURES REMAIN SUCH AS VIRTUAL MEMORY PAGING/SWAPPING
  - VRTX: NO MULTI-PROGRAMMING CAPABILITY
  - BiIN: UNFORTUNATELY, GOING OUT OF BUSINESS
- FY89 FINAL REPORT IS AVAILABLE: "ACQUIRING A REAL-TIME OPERATING SYSTEM FOR THE SPACE STATION FREEDOM", BY ROGER RACINE OF CSDL

# NOTES - CONCLUSIONS

Our analysis of commercial operating systems used vendor literature and our combined set of operating system requirements. The analysis found that the following operating systems came closest to meeting our set of requirements: Regulus, Galaxy, VxWORKS, Harris, DDC-I, and LynxOS.

The analysis also found that the following commercial operating systems were unsuitable: DEC VAXELN, CMU MACH, VRTX, and BiiN. VAXELN was determined unsuitable because it is only available on VAX hardware and it would likely be too costly to port it to the space station flight computer. The MACH operating system was determined unsuitable because it still contains many performance-harming features of Unix in its kernel such as virtual memory paging/swapping to mass memory. VRTX was found unsuitable because it does not support multi-programming. And finally, BiiN was found unsuitable because, unfortunately, the company is going out of business.

The FY89 final report on the results of this study is available. It is titled "Acquiring a Real-Time Operating System for the Space Station Freedom" and was written by Roger Racine of Charles Stark Draper Laboratory, Inc.

# FUTURE PLANS

- FY90

- DEVELOP DETAILED EVALUATION CRITERIA FOR DMS  
ADVANCED OPERATING SYSTEM
- DEVELOP BENCHMARK TEST SOFTWARE
- TEST AND EVALUATE CANDIDATE OPERATING SYSTEMS
- WRITE FY90 REPORT

- FY91

- IDENTIFY APPROACHES TO GRACEFUL EVOLUTION OF THE  
DMS OPERATING SYSTEM
- EVALUATE APPROACHES
- DEVELOP TRANSITION PLAN
- WRITE FY91 REPORT

# NOTES - FUTURE PLANS

The DMS Advanced Operating System Study plan for FY90 is to first, develop a set of detailed evaluation criteria and weighting factors. Secondly, benchmark test software will be developed to test important functions and their performance. Weighting factors will also be developed for the benchmark tests. The complete set of evaluation criteria and weighting factors will then be used to evaluate the most promising commercial candidate operating systems. Finally a report describing this work and its results will be written.

The plan for FY91 is to identify various approaches to the graceful evolution from the initial DMS Operating System to the advanced. These approaches will then be analyzed and evaluated and a transition plan will be developed. Finally a report describing this work and its results will be written.

# **SPACE STATION EVOLUTION**

## **Beyond the Baseline**

### **OMS Advanced Automation: FDIR Prototyping**

**Matthew A. Hanson  
Ford Aerospace Corp.  
1322 Space Park Dr.  
Houston, TX 77258**

## **ABSTRACT**

**The purpose of this project is to address the global fault detection, isolation and recovery (FDIR) requirements for OMS automation within the Space Station Freedom program. This shall be accomplished by developing a selected FDIR prototype for SSFP distributed systems. In particular, FDIR functionality for the management of ground and onboard computer networks will be prototyped (initially in JSC's SSCC test bed). The first of five incremental prototypes is currently in its coding phase, and will be demonstrated in January 1990. The final prototype is planned for a March 1992 demonstration. The prototype shall be based on advanced automation methodologies in addition to traditional software methods to meet the requirements for automation. A rule-based expert system driven by an object-oriented model of the physical network is being prototyped to accomplish this. The flow of the system executive and diagnostic process are overviewed, and the progression of the project through incremental prototypes is outlined. The intent is to develop a system that can be transitioned to operational status, and to identify hooks and scars to make it so. Another goal of this project is to establish an approach to distributed system FDIR that can be related to other station-wide fault management systems. In this light, the advantages and lessons-learned of the implementation will be discussed throughout the project via regular briefings and technical documentation.**

## **INTRODUCTION**

---

- **Personnel Profile**
  - **Code S RTOP : Gregg Swietek**
  - **JSC Monitor (SDD) : Catherine Williams, Mike Kearney (Network and Communications Systems Development Section)**
  - **Ford Aerospace : Matt Hanson, Eric Taylor, Charles Copeland, John Engvall**
  
- **The OMS is a set of automated & manual functions**
  - **Improve safety, reliability & productivity**
  - **Reduce maintenance & operations cost**
  
- **FDIR (Fault detection, isolation & recovery) is a major OMS functional area**
  - **Controlled mode intersystem FDIR**
  - **Trend Analysis**

# **INTRODUCTION**

**This is a Code S RTOP sponsored by Gregg Swietek at Headquarters. It has been contracted to Ford Aerospace Corp. in Houston, and locally monitored by Catherine Williams through Mike Kearney's Network and Communications Systems Development Branch (within the Systems Development Division). At Ford, it is principally manned by Matt Hanson (technical lead), Eric Taylor, Charles Copeland and John Engvall (Advanced Research Section Head). The Ford people are members of their AI Lab, which was established in 1983.**

**The Space Station Operations Management System (OMS) will automate major management functions to coordinate the operations of SSFP systems, elements and payloads. The objectives of OMS are to improve safety, reliability and productivity while reducing maintenance and operations cost. This will be accomplished by using advanced automation techniques to automate much of the activity normally performed by the flight crew and ground personnel. The OMS is a set of functions which includes application software and manual interactions with the Freedom Station either from onboard or on the ground. OMS requirements have been organized into twelve functional groups for both OMA and OMGA.**

**The scope of this prototyping effort falls within the Station-Wide Fault Management (SWFM) requirements group for OMS. Prototyping will center on the application of advanced automation techniques to computer network FDIR for ground and onboard computer networks. This selection will provide a real FDIR scenario (vs. a simulation) that will provide valuable experience in advanced automation applied to distributed system FDIR. The prototypes will operate via "controlled mode," giving the user the various levels of override and general control required of OMS FDIR.**

## **BACKGROUND**

---

- Distributed processing is replacing centralized processing for ground and onboard systems
  - MCCU architecture
  - SSFP DMS and SSCC
- Prototype Application Focus: Computer Network Management FDIR
  - For onboard and ground based networks
  - Utilizing actual systems (vs. simulations)
  - Demonstrate system working across test beds
- Global FDIR impact
  - The health and status of the distributed computer networks is key to the functionality of complete FDIR functionality
  - Prototype a system whose design can be abstracted and applied to the general problem of FDIR for distributed SSFP systems

## **BACKGROUND**

**Distributed processing is replacing the central processing/dumb terminal architecture for command and control. The NSTS Mission Control Center Upgrade, the DMS and SSCC architectures are all representative of this trend. This evolution presents many advantages, but adds the complication of managing the distributed processing across these networks.**

**Fault Detection, Isolation and Recovery for these networks is the application focus of this prototyping effort. Since the target environment physically exists, valuable experience will be gained in applying advanced automation to FDIR on distributed systems. Our goal is to demonstrate this application and its architecture on JSC's SSCC and DMS test beds.**

**Computer network FDIR is integral to global FDIR. Many onboard system problem scenarios will be directly attributable to failure of computer network components. The primary functional goal of this prototype is to demonstrate rapid FDIR capability that is sensitive to the mission safety and mission success priorities of individual processing nodes. This application would aid and serve the network administrator. Another major goal is the design and demonstration of a knowledge-based system architecture that can be generally applied to FDIR applications for distributed systems.**

## **APPROACH**

- Assume FDDI backbones for ground and on-orbit computer networks
  - Onboard subnets token passing bus or token passing ring
  - Currently assuming ethernet subnets for SSCC
- Select highly capable, low risk advanced automation methods and tools
  - User I/F implemented using X-Windows
  - OOP used to model Physical system configuration and dynamic status (using C + + )
  - Rule-based KBS (Ops83) performs FDIR heuristics on OOP model
- Initial prototyping will use simulation of physical network topology
  - Simulation capability will continue to be developed as a training aid and demonstration tool

# **APPROACH**

**In our approach we target an FDDI (Fiber Distributed Data Interface, 100 Mbps token ring) backbone with various subnet topologies. These subnets will be token passing busses or token passing rings to represent the DMS, and ethernetets to represent a likely SSCC scenario.**

**Through our initial prototype baseline activity, we have selected an architecture and tool set. The architecture is overviewed on the next slide. This slide lists the tools we've selected. The system is being developed on a Sun workstation, with the objective of making it Unix-transportable. MIT's X-Windows will be used as the user interface for windows, graphics, and user event handling. X-Windows has broad industry support as a de-facto standard. Our design employs object oriented programming to model the network topologies. The rationale for the models will be explained on the next slide. C++ is being used to implement these models. The production systems are being developed using Production Systems Technologies' Ops83. Ops83 is the fastest production system we've bench-marked, features a programmable recognize-act cycle, and interfaces readily to the rest of the selected tools. Hardware vendor independence is highly desirable, and applications developed using these tools have proven to be readily transportable across Unix workstations.**

**The initial prototype will simulate the physical network as a dynamic data source for the system. Just as in aircraft simulators, this approach will allow training for difficult scenarios without actually "crashing" anything. Beyond training, the simulator will make the KBS heuristics development simpler, and will provide for effective demonstrations. The initial prototype is scheduled for mid-January.**

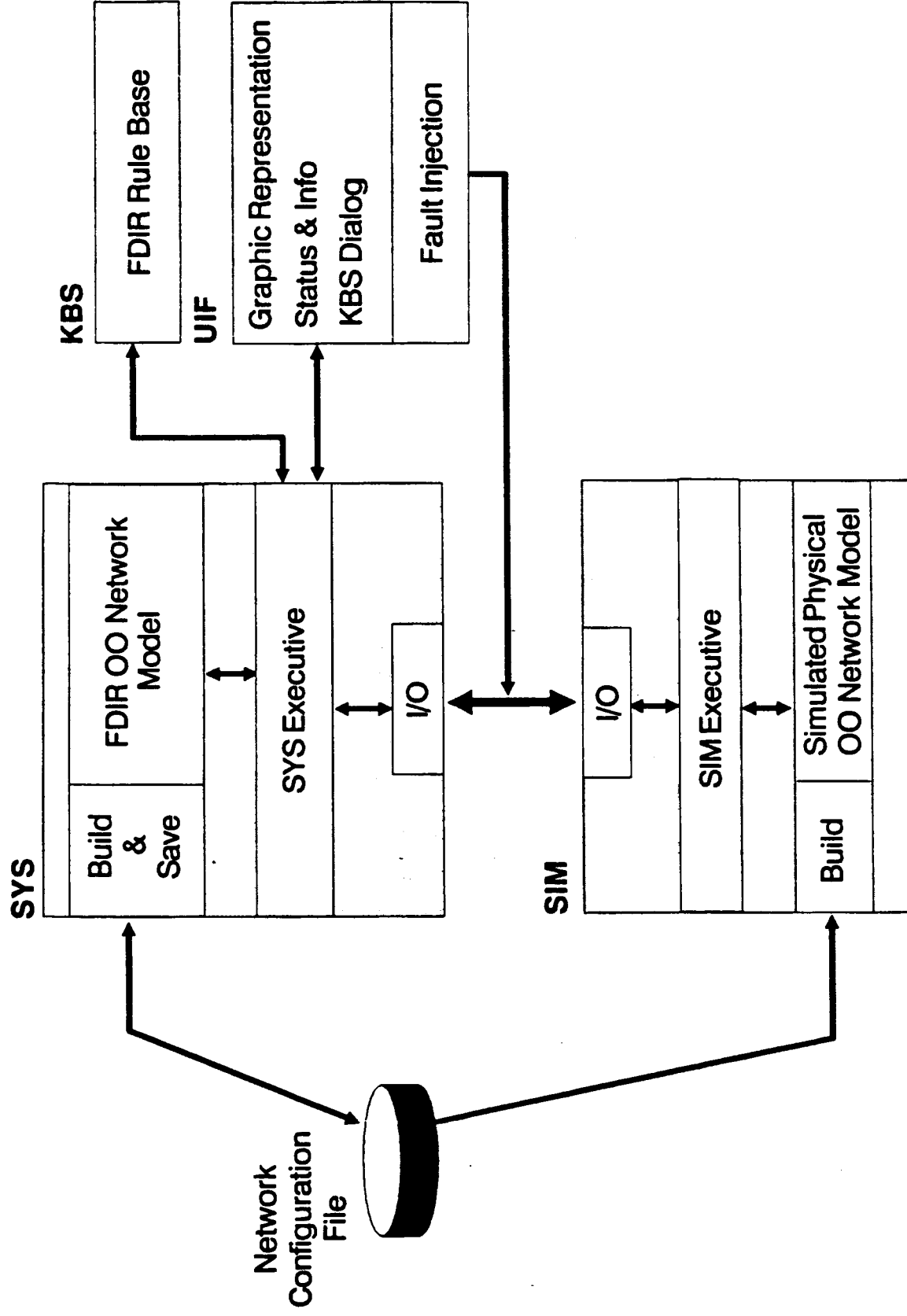
## **APPROACH (cont.)**

- **Intermediate prototypes will utilize actual SSCC testbed network components**
  - **Will give insight to problems associated with interacting with real systems**
  - **Make/buy vendor network management capability to support this**
  - **Continue to expand simulation capability**
  
- **Later prototypes will continue to extend this functionality**
  - **Incorporate SSFP-specific network FDIR scenarios**
  - **Demonstrate extension of architecture to station-wide fault management.**

## **APPROACH (cont.)**

Four more incremental prototypes will follow the first at approximately 24-week intervals. The intermediate prototypes will interact with network hardware on JSC's SSCC test bed. A Fibronics FDDI system is currently in place on this test bed, and we are currently investigating COTS solutions to provide the KBS with the monitoring capability and control authority desired. The simulator will continue to be refined throughout the project. In addition, a major emphasis will be to add robustness to the KBS heuristics during development of the second, third, and possibly fourth prototypes.

The fourth and fifth (final) prototypes will continue to add robustness to the simulator and to the KBS's FDIR capability. For these, SSFP-specific FDIR scenarios will be developed and demonstrated as SSFP operations concepts mature. This will involve recovery steps sensitive to processing node priorities concerning mission safety and mission success. We also have a goal of interacting with other OMS prototyping efforts. For example, the prototype could provide feedback to planning functions concerning network reconfiguration scenarios.



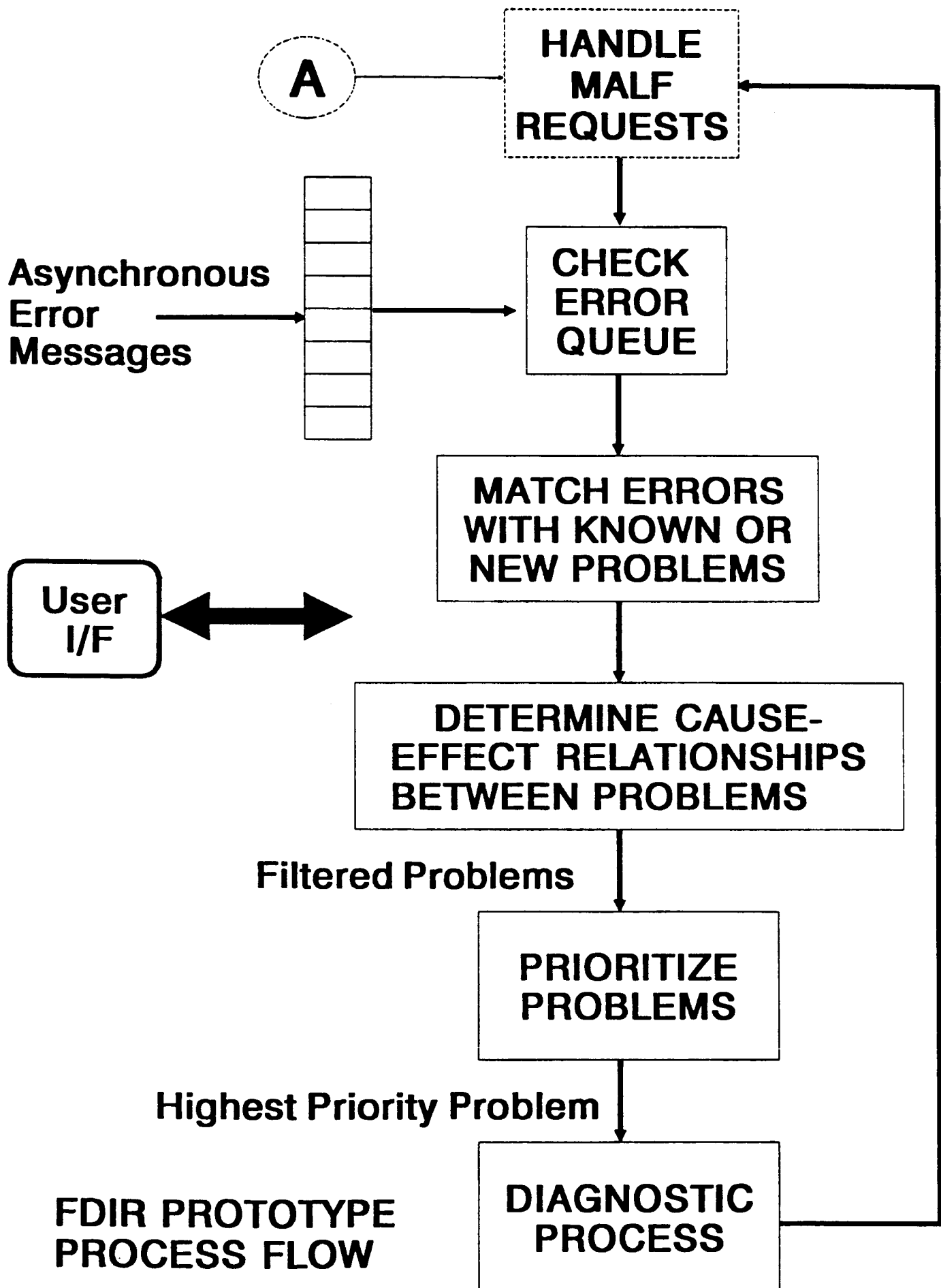
**Proto-1 Block Diagram**

## **APPROACH (cont.) - PROTOTYPE-1 OVERVIEW**

**This figure shows an overview of the first prototype. The SYStem process contains the executive and keeps an object- oriented model of the physical network at hand for KBS processes to perform FDIR heuristics on. The executive spawns all other processes, and is the "heartbeat" of the system. The process flow that describes this heartbeat is illustrated on the following slide. The OOP model of the physical network contains a facsimile of the basic network configuration and dynamic status of the network. Dynamic status is updated through unsolicited error messages from network nodes and as-needed polling of the network nodes. Polling demands may come from KBS processes or the network administrator.**

**The SIMulation process is similar to the SYS OOP model. This is the network malfunction simulator. System fault requests from the user interface will be translated to symptoms and error messages that would be visible to the executive through OSI network management services and protocols.**

**The KBS and User Interface processes are shown interacting with the executive. This interaction is described in the high level process flow on the following slide. For KBS development and demonstration, the "game" of the system will be to correctly diagnose the fault that was "injected" into the SIM model based on error messages and retrievable system parameters. Having achieved this, the next step would be to recommend a recovery plan. All or part of the plan may be executed automatically depending on the control mode and the existence of control mechanisms.**



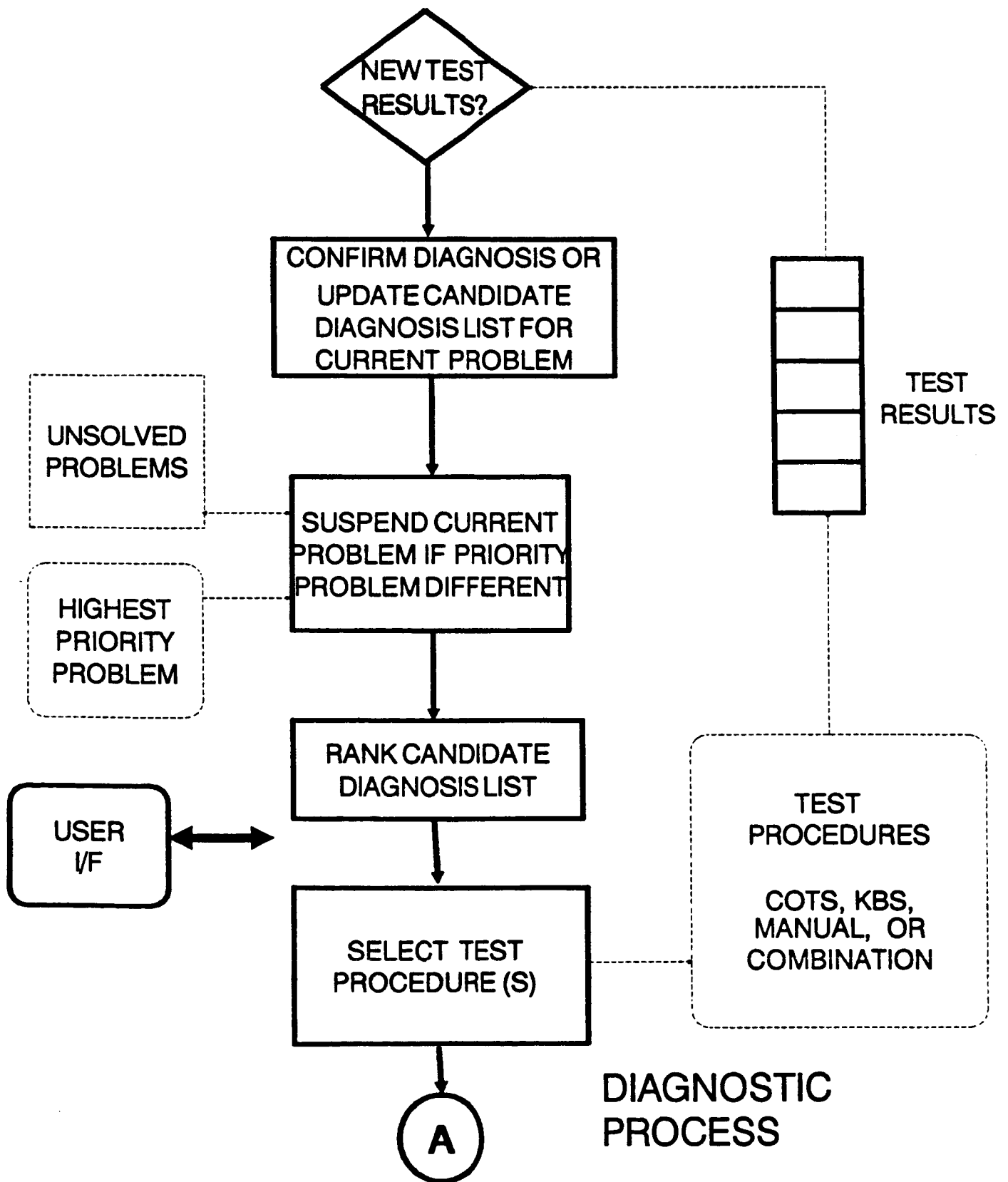
## **APPROACH (cont.) - PROTOTYPE PROCESS FLOW**

**This figure describes the system heartbeat referenced in the previous slide. The User I/F process is shown interacting with all stages of the process flow.**

**The SIM model first handles malfunction requests from the user. The concurrent processing of the SIM is not shown here, but suffice to say that it inserts the malfunctions into the model. The SIM objects respond by having their member functions simulate the symptoms of the malfunction. Some of these symptoms are asynchronous error messages sent to the SYS.**

**The error message queue is checked for new messages, which are then heuristically mapped to "problems." Heuristics are then used to organize the problem instances into cause-effect relationships. The purpose is to isolate root problems for diagnosis.**

**The importance and severity of each problem is determined, and the highest priority problem is passed to the Diagnostic Process. The priority of the problem may also be a function of its solvability. If test results are required for further diagnosis, the priority may be lowered until the results are available. The Diagnostic Process, shown as the last box in the cycle, is expanded on the following slide.**



## **APPROACH (cont.) - DIAGNOSTIC PROCESS**

**Before analyzing the highest priority problem passed in, the Diagnostic Process takes care of any unfinished business. The test results queue is checked to see if there are any new results ready from previous requests to run test procedures. If there are (and if they apply to the problem the KBS is currently working on), the KBS uses them to draw conclusions about its list of potential diagnoses. It may confirm the diagnosis and notify the network administrator of recovery recommendations, or it may add or remove candidate diagnoses from its list for this problem.**

**Note that the test results may involve problems that are currently "suspended." These problems may receive a higher priority now that more data is available.**

**The highest priority problem is now compared to the current problem. If different, the current problem is "suspended" in favor of the higher priority problem. A list of candidate diagnoses is generated if this is a new problem, and the list is ranked by likelihood.**

**Test procedures are then heuristically selected, to confirm or deny candidate diagnoses. Note that these test procedures may range from simple status gathering to employing technicians to carry out complex procedures.**

**The "A" exiting the last box returns control back to the top of the Process Flow. Note that this Diagnostic Process assumes that only one problem is being diagnosed at any given time. This concept can be extended by spawning concurrent diagnostic processes. We are currently evaluating this mechanism for possible inclusion in future prototypes.**

## **RESULTS ACHIEVED/EXPECTED**

- **Baseline Study established prototyping functional goals and hosting decisions**
  
- **First of five prototypes scheduled for late-January demos**
  - **Establish look-and-feel & major functions of user interface**
  - **Establish "SYS" and "SIM" models**
  - **Demonstrate KBS approach**

## **RESULTS ACHIEVED/EXPECTED**

**The front-end of this project consisted of several trade studies encapsulated as the Baseline Study. Through these studies, we formulated the initial prototype application focus, hardware and software hosting decisions, and projected the path of the remaining prototypes. We regret the title of this tech note, as it has been misconstrued as an attempt to baseline OMS FDIR requirements.**

**As of this writing, we are right in the middle of the first prototype's coding cycle. After it is demonstrated in late January (1990), we plan to publish a tech note further detailing the design, lessons learned, and the functional goals of the remaining incremental development.**

**The goals of the first prototype are to completely shell the user interface and establish its major functions, to establish the SYS and SYM object-oriented models and their primary member functions, and to integrate the rule-based production system. We plan to have at least one or two FDIR scenarios built in for the purposes of evaluating our production system interface and demonstration.**

## **BASELINE IMPACTS, DESIGN REQUIREMENTS & TRANSITION PLANS**

- **Unix and OSI are targeted as the prototype operating environment**
- **User I/F is written in X Windows**
- **Advanced automation tools are Unix transportable**
  - **No SSFP standardization as yet**
  - **Rule-based KBS and OOP approach are baselined as SSE KBSE criteria**
- **Network Architecture (H/W and S/W) would need to converge with KBS**
  - **KBS interfaces will utilize OSI standards**
  - **KBS heuristics and OOP models would require application specific tuning**
- **Processing resource requirements for implementation strategy will be discussed**

## **BASELINE IMPACTS, DESIGN REQUIREMENTS & TRANSITION PLANS**

**In choosing our prototyping tools and environment, we have attempted to balance the need to use tools that allow us to prototype rapidly, with the desire to wind up with a system that may be transitioned into operations. In choosing tools, we constrained ourselves to "conservative" advanced automation technologies that are supported by SSFP documentation. In particular, rule-based programming and object-oriented programming are identified as SSE ESDE (Expert System Development Environment) attributes. The actual ESDE is as yet undefined, however.**

**We anticipate that many of our computer network FDIR scenarios will be general purpose, requiring little adaptation. However, there will also be several needing adaptation to actual onboard and ground computer network topologies and operations concepts. Similarly, we plan to flag computer network design requirements for this type of advanced automation as we uncover them. We have baselined the general topologies and standards that we believe will be applied to SSFP, though the standards and designs are still moving targets. We anticipate that this approach will minimize both the hooks-and-scars requirements and transition plans as much as possible, though they will most likely still be considerable efforts. As these impacts clarify during the prototyping process, they will be addressed in our regular status reports and documentation.**

## **CONCLUSION**

- **Incremental prototyping effort for advanced automation applied to OMS FDIR**
- **Application focus: SSFP computer network FDIR**
  - **Onboard and ground networks**
- **A prototype architecture for advanced automation applied to station-wide fault management**
- **Establish plan to transition to operational system**

## **CONCLUSION**

**This is a prototyping project to investigate the application of advanced automation techniques to the OMS FDIR requirements group. The application focus for this prototyping effort is SSFP computer network FDIR. The system is being built with the controlled mode requirements specified for OMS FDIR functions. Such a system would be a valuable companion to the network administrator in more efficiently detecting, isolating and recovering from malfunctions. This will result in increased mission safety and success, and lower operating costs.**

**This application can be abstracted to performing FDIR on a distributed system. Therefore, its advantages and disadvantages will be known as other station-wide fault management systems are designed and developed.**

**Our goal is to prototype a system that can be transitioned to operational status. In addition, we are building in the capability for the system to perform as a training device. From an operating system, OSI standards and network topology viewpoint, the system should be adaptable to SSFP computer network environments. The heuristics built into the knowledge based system will probably require a moderate transition effort. We plan to minimize this by interacting with SSFP design and operations concepts efforts.**

**OPERATIONS  
MANAGEMENT  
SYSTEM**

richard e. eckelkamp  
mission planning  
and analysis

february 8, 1990

***Event Evaluator  
and  
Scheduler***

a b  
l a  
e r m  
x r d  
m a y a  
i n c  
k d f  
e e o  
r x

## **OUTLINE**

**BACKGROUND**

**STATION OPS ENVIRONMENT**

**OMS PROCESS CONTROL**

**RESULTS**

**STRATEGY**

**FUTURE GOALS**

**STATION IMPACTS**

**CONCLUSION & CHALLENGE**

## **BACKGROUND**

**OBJECTIVE : TO PROVIDE AN ORDERLY & EFFICIENT MANNER OF BUILDING, OPERATING, & MAINTAINING THE SPACE STATION.**

**METHOD : MAKE USE OF AN OPERATIONS MANAGEMENT SYSTEM THAT**

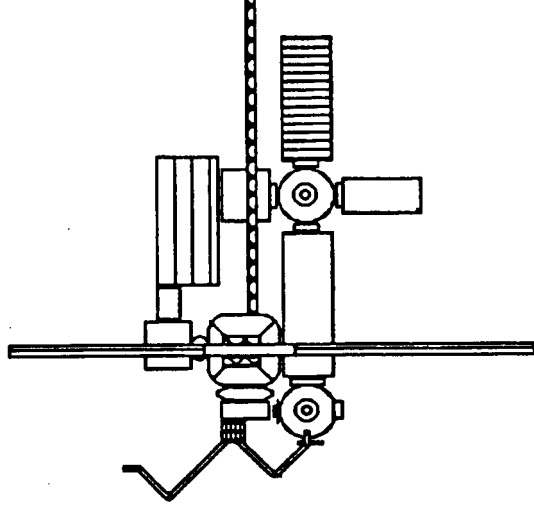
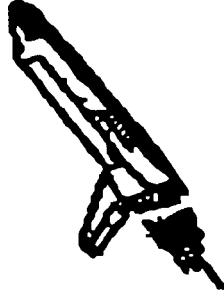
**PROVIDES INTEGRATED STATUS, COMMAND, & CONTROL  
USES DATA DRIVEN SHORT TERM PLAN TO ALLOW  
FLEXIBILITY OF OPERATIONS  
IS BUILT TO ALLOW EVOLUTION OF FUNCTION & FORM.**

**THE EVENT EVALUATOR PROTOTYPES ESSENTIAL PARTS OF  
THE SHORT TERM PLAN EXECUTION PROCESS.**

# ***OMS***

---

## **STATION - A NEW OPERATIONAL ENVIRONMENT**



**SHORT, OCCASIONAL FLIGHTS**

**CONTINUOUS MANNED PRESENCE**

**MOSTLY GROUND MAINTENANCE**

**ROUTINE ONORBIT MAINTENANCE**

**GROUND TESTING**

**ONORBIT TESTING**

**DISCRETE PREMISSION PLANNING  
& REALTIME CONTROL CENTER  
MODIFICATIONS DONE MANUALLY**

**CONTINUAL PLANNING & EXECUTION  
LINKED PROCESS WITH SIGNIFICANT  
AUTOMATED ASSISTANCE**

## **ANALOGY**

**SHUTTLE :**

**MAKING A SINGLE ITEM  
OR A SINGLE VIAL OF CHEMICAL  
BY HAND**

**STATION :**

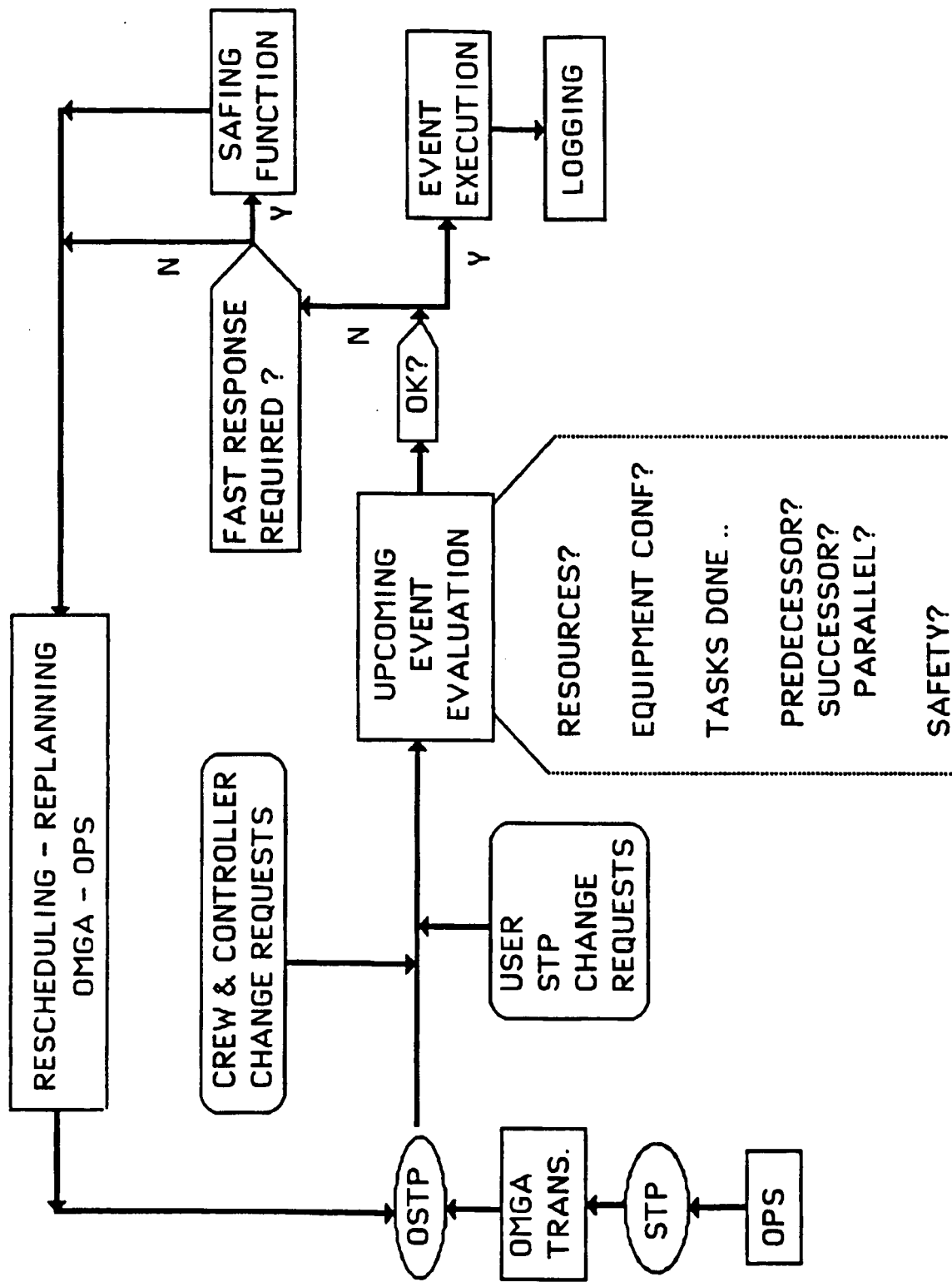
**USING AN ASSEMBLY LINE -> OBJECT  
OR A REFINERY -> CHEMICAL  
IN A PROCESS - CONTROL ENVIRONMENT**

**PROCESS CONTROL BY ITS NATURE :**

**COMPENSATES FOR PRODUCTION /OPERATIONS VARIATIONS  
REACTS TO PROBLEMS  
CAN BE TUNED TO ACHIEVE OBJECTIVES**

## OMS OPERATIONS CONTROL PROCESS

5-4-89

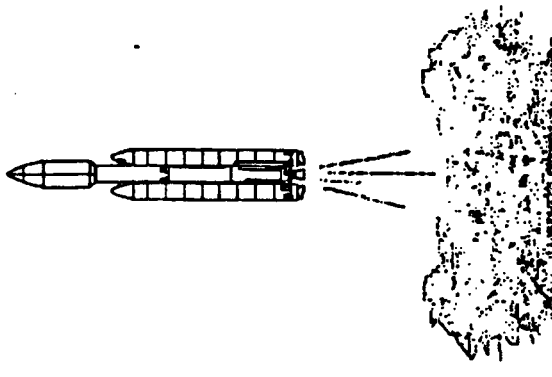


## **RESULTS**

**WHAT : SCHEDULED EVENTS EVALUATION &  
RESOLUTION (SEER) PROGRAM BUILT  
AS PART OF COMPUTER - AIDED SCHEDULING  
PROGRAM (COMPASS) .**

### **CHARACTERISTICS :**

- \* SOFTWARE ENGINEERED - IN ADA WITH CALLS TO C - ENCASED  
X - WINDOWS USER INTERFACE**
- \* MODULAR - ASSEMBLED FROM 26 GENERIC PACKAGES OR  
UNITS, EACH COMPOSED OF MULTIPLE PROCEDURES  
& FUNCTIONS, DEVELOPED & VERIFIED BEFOREHAND**



## **RESULTS (CON 'T)**

### **CHARACTERISTICS :**

- \* PORTABLE - AVAILABLE ON**

**SUN SPARC, 3/SERIES USING VERDIX**

**APOLLO DM 3100, 3010, 3500 (SOFTWARE**

**SUPPORT ENVIRONMENT MACHINES) USING ALSYS**

**RATIONAL**

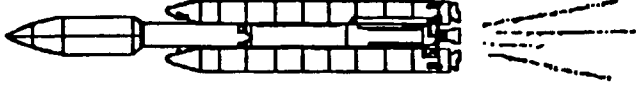
**BEING PORTED TO PS-2 MACHINE**

- \* FLEXIBLE - DATA-ORIENTED DESIGN MAKES IT A**

**GENERIC PROGRAM , ABLE TO HANDLE MANY  
APPLICATIONS.**

- \* SMALL - 12000 LINES OF CODE FOR EVENT EVALUATOR  
SCHEDULER, AND USER INTERFACE**

- \* AVAILABLE - PROGRAM OR INDIVIDUAL UNITS  
FOR THE ASKING**



## **STRATEGY**



- 1 ) PRODUCING THE EVENT EVALUATOR AS PART OF A SCHEDULER**
  - A) REDUCED CODING COSTS**
  - B) ENABLED THE NATURAL LINKAGE BETWEEN THE TASK THAT FINDS SCHEDULE PROBLEMS WITH THE TASK THAT CORRECTS THE SCHEDULE.**
- 2 ) USE OF STATION STANDARD SOFTWARE ENGINEERING PRACTICES, COUPLED WITH 1) ABOVE MAKES SEER-COMPASS A GOOD CANDIDATE FOR THE FLIGHT OMS ( ON BOARD & GROUND ).**

## **MORE RESULTS**

**CURRENT PROGRAM HANDLES :**

**TIME CONSTRAINTS - REQUIRED INTERVALS**

**DURATION**

**PREDECESSORS**

**SUCCESSORS**

**RESOURCE CONSTRAINTS**

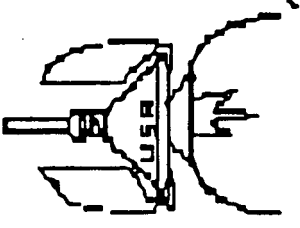
**PRIORITIES**

**TO BE ADDED THIS YEAR :**

**PREFERRED INTERVALS**

**BOOLEAN & OTHER RULE - BASED CONSTRAINTS**

**MULTIPLE PARALLEL PROCESSES**



## MORE RESULTS (CON'T)



**DATA SETS HAVE BEEN DEVELOPED FOR :**

**EXHAUSTIVE VERIFICATION OF CAPABILITIES**

**GENERALIZED SKYLAB EXPERIMENTS**

**SELECTED STATION CORE ACTIVITIES**

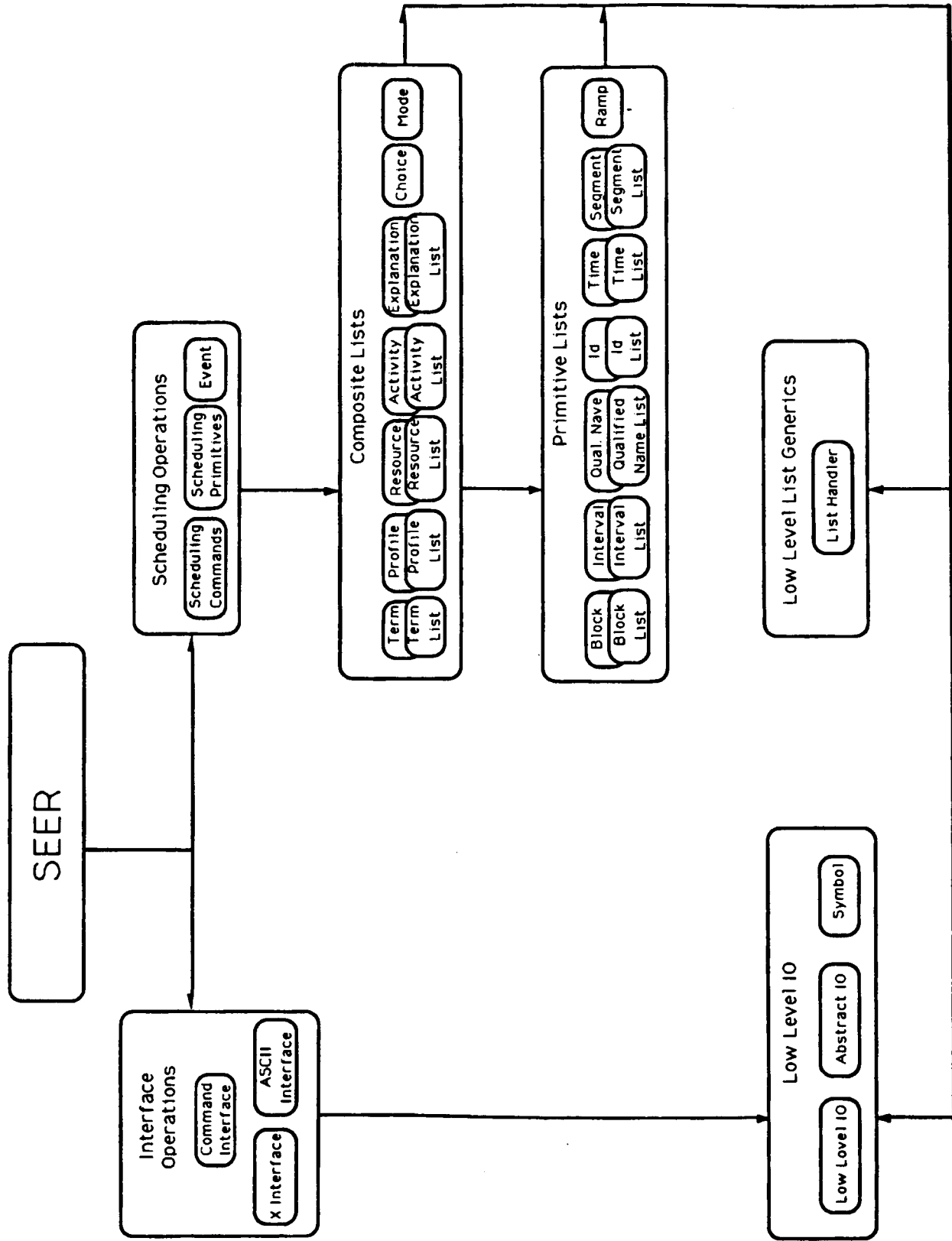
**TYPICAL RELATIVE PERFORMANCE DATA :**

**straight forward option ( schedule as soon as possible)**

**on SUN 3/60 ( slower than a PS-2)**

**5 resources, 4 time constraints, priorities**

# OF EXPERIMENTS	# OF ACTIVITIES	CPU TIME, SECS DATA LOAD	SCHEDULE
10	34	9	11
20	68	14	24
40	136	35	64
80	272	63	178



# COMPASS

1.0000000000000000  
2.0000000000000000  
3.0000000000000000

4.0000000000000000  
5.0000000000000000  
6.0000000000000000

7.0000000000000000  
8.0000000000000000  
9.0000000000000000  
10.0000000000000000  
11.0000000000000000  
12.0000000000000000

13.0000000000000000  
14.0000000000000000  
15.0000000000000000

16.0000000000000000  
17.0000000000000000  
18.0000000000000000  
19.0000000000000000  
20.0000000000000000

21.0000000000000000  
22.0000000000000000  
23.0000000000000000

24.0000000000000000  
25.0000000000000000  
26.0000000000000000  
27.0000000000000000

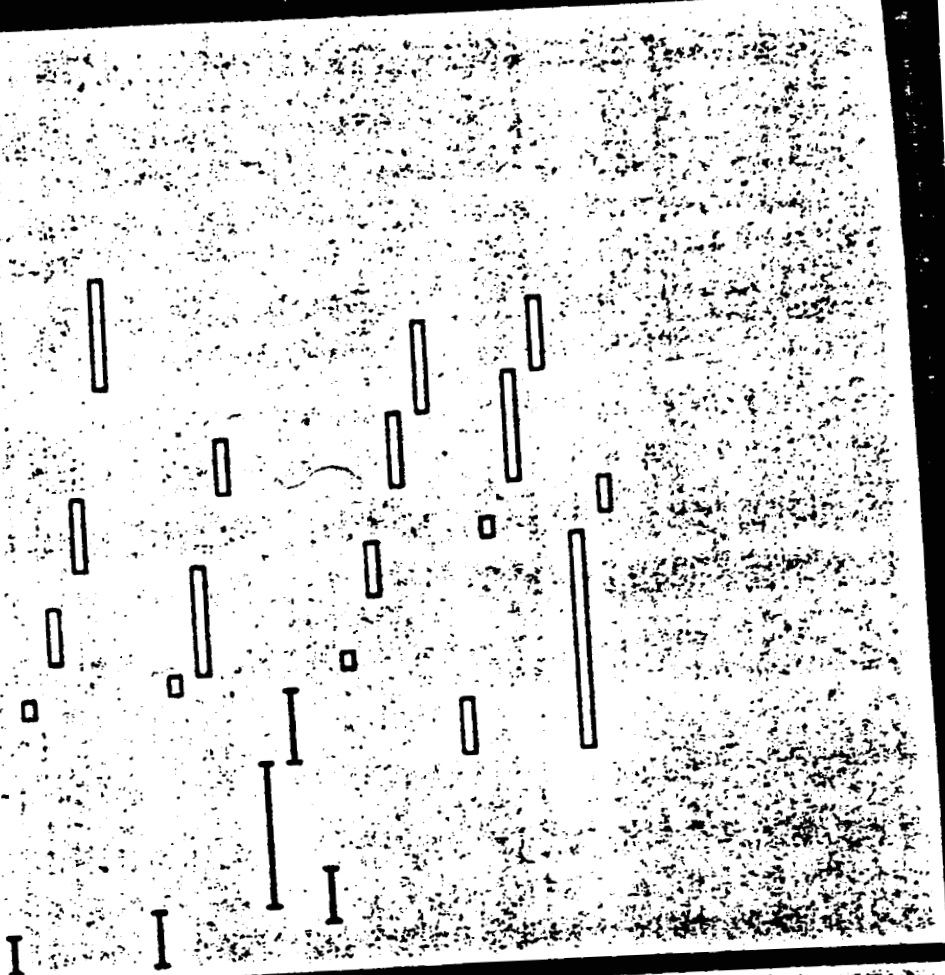
28.0000000000000000  
29.0000000000000000  
30.0000000000000000

31.0000000000000000  
32.0000000000000000  
33.0000000000000000

34.0000000000000000  
35.0000000000000000  
36.0000000000000000

exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. technology\_2  
exp\_3. life\_science  
exp\_3. life\_science  
exp\_4. life\_science  
exp\_4. life\_science  
exp\_4. life\_science  
exp\_4. life\_science  
exp\_5. space\_plasma  
exp\_5. space\_plasma  
exp\_5. space\_plasma  
exp\_5. space\_plasma  
exp\_5. astronomy\_2.1  
exp\_5. astronomy\_2.1

37.0000000000000000  
38.0000000000000000  
39.0000000000000000  
40.0000000000000000  
41.0000000000000000  
42.0000000000000000  
43.0000000000000000  
44.0000000000000000  
45.0000000000000000  
46.0000000000000000  
47.0000000000000000  
48.0000000000000000  
49.0000000000000000  
50.0000000000000000  
51.0000000000000000  
52.0000000000000000  
53.0000000000000000  
54.0000000000000000  
55.0000000000000000  
56.0000000000000000  
57.0000000000000000  
58.0000000000000000  
59.0000000000000000  
60.0000000000000000  
61.0000000000000000  
62.0000000000000000  
63.0000000000000000  
64.0000000000000000  
65.0000000000000000  
66.0000000000000000  
67.0000000000000000  
68.0000000000000000  
69.0000000000000000  
70.0000000000000000  
71.0000000000000000  
72.0000000000000000  
73.0000000000000000  
74.0000000000000000  
75.0000000000000000  
76.0000000000000000  
77.0000000000000000  
78.0000000000000000  
79.0000000000000000  
80.0000000000000000  
81.0000000000000000  
82.0000000000000000  
83.0000000000000000  
84.0000000000000000  
85.0000000000000000  
86.0000000000000000  
87.0000000000000000  
88.0000000000000000  
89.0000000000000000  
90.0000000000000000  
91.0000000000000000  
92.0000000000000000  
93.0000000000000000  
94.0000000000000000  
95.0000000000000000  
96.0000000000000000  
97.0000000000000000  
98.0000000000000000  
99.0000000000000000  
100.0000000000000000



COMPASS

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

```
name exp_1,technology_2,operation_3,step_2
scheduled t
valid_duration t
valid_interval t
invalid_predecessors {}
invalid_successors {}
invalid_assignments {}

name exp_1,technology_2,operation_3,step_3
scheduled t
valid_duration t
valid_interval t
invalid_predecessors {}
invalid_successors {}
invalid_assignments {}

name exp_1,technology_2,operation_3,step_4
scheduled t
valid_duration t
valid_interval t
invalid_predecessors {}
invalid_successors {}
invalid_assignments {}

name exp_1,technology_2,operation_3,step_5
scheduled t
valid_duration t
valid_interval t
invalid_predecessors {}
invalid_successors {}
invalid_assignments {}

name exp_2,technology_3,operation_3,step_3
scheduled t
valid_duration t
valid_interval t
```

COMPASS 1.0.0



SHRINK DISPLAY

HORIZONTAL SCROLL

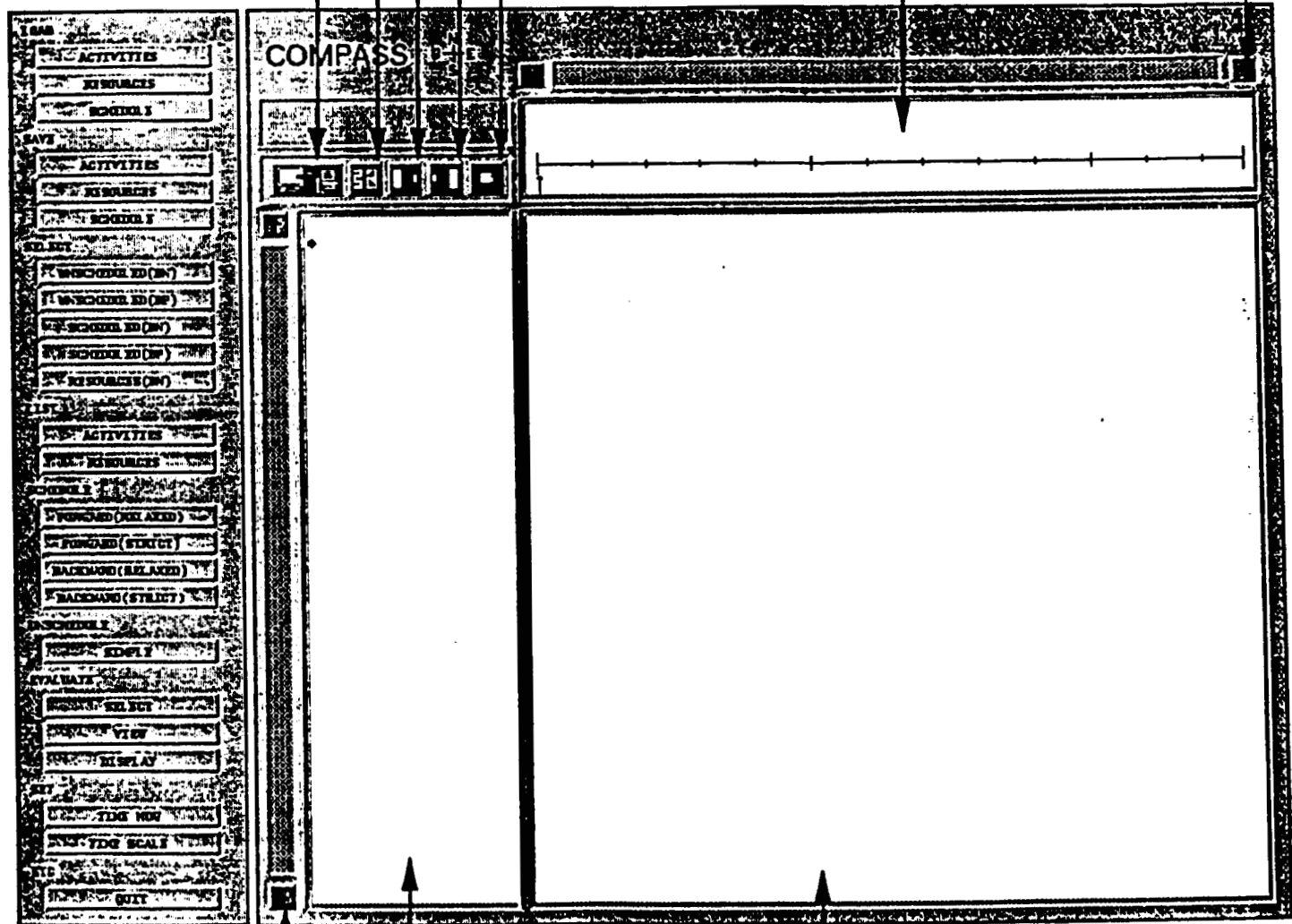
ACTIVITY/RESOURCE  
DISPLAY

EXPAND DISPLAY

TIME SCALE  
WINDOW

SCREEN PRINT

ADJUST  
SCROLLING/ZOOM



ACTIVITY/RESOURCE  
NAME WINDOW

GANTT-CHART/RESOURCE  
WINDOW

VERTICAL SCROLL

COMMAND WINDOW



## **FUTURE GOALS**

**MODIFY SEER-COMPASS TO HANDLE BOOLEAN & OTHER CONSTRAINTS  
DETERMINE METHODS OF MODELING STATION OPERATIONAL STATES**

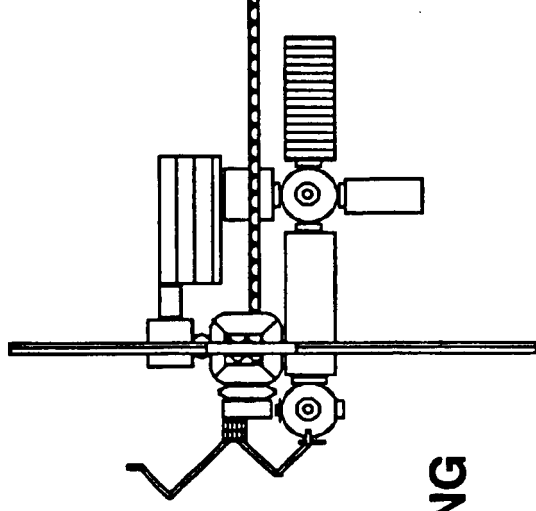
**TEST VARIOUS SCHEDULING TECHNIQUES OR HEURISTICS  
DEVELOP METHODS OF PREDICTING FUTURE OPERATIONAL STATES**

**DEFINE INTERFACES AMONG SEER-COMPASS, THE PLANNING SYSTEM,  
AND REALTIME MONITOR & CONTROL TASKS  
TEST SEER-COMPASS IN THE JSC OMS-DMS TESTBED**

**DEVELOP DETAILED MODELS FOR DESCRIBING RESOURCE USAGE  
APPEND OMS EVOLUTION DOCUMENT TO :**

**OUTLINE THE ACTIVATION PLAN FOR OMS DURING ASSEMBLY  
UPDATE PLAN TO INTRODUCE AUTOMATION**

## **STATION IMPACTS**



**ALTHOUGH SUFFICIENT TESTS INVOLVING  
STATION DATA SETS HAVE NOT BEEN  
PERFORMED, IT APPEARS THAT AN  
ONBOARD COMPUTER, FASTER THAN THE  
PS-2, MIGHT BE IN ORDER TO REDUCE  
EVENT EVALUATION & RESCHEDULING  
CPU TIME --->> NO SHOW STOPPER.**

## **CONCLUSION AND CHALLENGE**

**AN ADA EVENT EVALUATOR, COUPLED WITH A SCHEDULER, HAS BEEN BUILT & IS BEING SUCCESSFULLY TESTED.**

**TASKS FOR FY90 WILL CONCENTRATE ON AREAS OF UNCERTAINTY IN THE OMS DESIGN.**

### **OUR CHALLENGE :**

**TO PERFORM CAREFUL RESEARCH &  
TO PROVIDE CONCRETE DESIGNS, BASED ON THE STATION WE  
ARE BUILDING,  
THAT ENABLE THE USE OF ADVANCED TECHNOLOGIES NOW  
& IN THE FUTURE  
THAT ALLOW OPERATORS TO ACHIEVE MORE EFFICIENT & SAFER  
OPERATIONS.**

**AUTOMATED SOFTWARE DEVELOPMENT WORKSTATION (ASDW)**

**Ernest M. Fridge III**

**NASA/JSC Software Technology Branch FR5**

## **ABSTRACT**

Software development is a serious bottleneck in the construction of complex information systems development and evolution. The heaviest development cost tends to occur in the early part of the life cycle during requirements generation, requirements analysis, design, and application development. Maintenance cost are even more. An increase of the reuse of any of the software "parts" used in these activities has been viewed as a way to relieve this bottleneck. One approach to achieving software reusability is through the development and use of software parts composition systems. A software parts composition system is a software development environment comprised of a parts description language for modeling parts and their interfaces, a catalog of existing parts, a composition editor that aids a user in the specification of a new application from existing parts, and a code generator that takes a specification and generates an implementation of a new application in a target language. The Automated Software Development Workstation (ASDW) is currently an expert system shell that provides the capabilities required to develop and manipulate these software parts composition systems. The ASDW is now in Beta testing at the Johnson Space Center. Future work centers on responding to user feedback for capability and usability enhancement, expanding the scope of the support for collecting, representing, and manipulating knowledge during the early phases of the information system lifecycle, and in providing solutions for handling very large libraries of reusable components.

## INTRODUCTION

### Goals

The ASDW (Advanced Software Development Workstation) task is researching and developing the technologies required to support CASE (Computer Aided Software Engineering) with the emphasis on those advanced methods, tools, and processes that will be required to support all of NASA. Immediate goals are to provide research and prototype tools that will provide near term productivity increases in projects such as the SSE (Software Support Environment), the SSCC (Space Station Control Center), and FADS (Flight Analysis and Design System) which is used to support the STS. Goals also include providing technology for future SSE and operational systems by adding knowledge based system support to all phases of information systems development, evolution, maintenance, and operation.

## **INTRODUCTION**

### **Goals:**

- \* RESEARCH AND DEVELOP KNOWLEDGE BASED TECHNIQUES FOR REUSE OF SOFTWARE ARTIFACTS TO SIGNIFICANTLY IMPROVE THE PRODUCTIVITY IN THE DEVELOPMENT, OPERATIONS, AND MAINTENANCE OF INFORMATION SYSTEMS.**
- \* REDUCE THE EFFORT INVOLVED IN BUILDING SOFTWARE PARTS COMPOSITION SYSTEMS**
- \* RESEARCH AND DEVELOP THE "FRAMEWORK" FOR ACQUIRING, REPRESENTING, INTEGRATING, AND TRANSLATING THE KNOWLEDGE REQUIRED TO DEVELOP AND OPERATE INFORMATION SYSTEMS**
- \* RESEARCH AND DEVELOP AN INTEGRATION "PLATFORM" PROVIDING COMPUTER AIDED SOFTWARE ENGINEERING (CASE) METHODOLOGIES, TOOLS, AND CONCEPTS FOR DEVELOPING INTEGRATED INFORMATION SYSTEMS**

## INTRODUCTION (CONTINUED)

### Background

Phase I (October, 85 to April, 87) demonstrated the feasibility of a knowledge based approach to application generation in a limited domain. Phase II (April, 87 to February, 89) investigated ways to exploit the use of knowledge representation, retrieval, and acquisition techniques. A prototype demonstrated a knowledge based system for the development of software parts composition systems (ie, a software parts composition shell). Phase III (March, 89 to December, 89) prototyped ways to handle the scale-up problem (1000-100000 objects), prototyped ways to automatically generate the taxonomy, and initiated two Beta test projects at JSC. One test project is to generate trajectory mission planning simulations from software parts and the other is to provide expert assistance to operational users in setting up input data for simulations. The projects will support the SSCC and FADS. During Phases 2 and 3, the project also began joint activity with the USAF studying the information requirements of information systems, their integration and development processes, the methodologies required, and the integrated tools and integrated knowledge base that supports this development. The USAF is providing around \$300K annually to study the modeling, methodologies, information requirements of manufacturing information systems, and to develop prototype tools. Modeling is based on the USAF's IDEF methodologies. JSC added funds to expedite the development of two methodology tools and a computer assisted tutor to educate developers in the proper use of the methodologies.

## **INTRODUCTION (CONTINUED)**

### **Background**

- \* PHASE I (10/85-4/87) PROTOTYPE DEMONSTRATED THE FEASIBILITY OF A KNOWLEDGE BASED APPROACH TO APPLICATION GENERATION IN A LIMITED DOMAIN**
- \* PHASE II (4/87-2/89) INVESTIGATED WAYS TO EXPLOIT THE USE OF KNOWLEDGE REPRESENTATION, RETRIEVAL, AND ACQUISITION TECHNIQUES. A PROTOTYPE DEMONSTRATED A KNOWLEDGE BASED SYSTEM FOR THE DEVELOPMENT OF SOFTWARE PARTS COMPOSITION SYSTEMS (IE, A SOFTWARE PARTS COMPOSITION SHELL).**
- \* PHASE III (2/89-12/89):**
  - \* PROTOTYPING WAYS TO HANDLE THE SCALE-UP PROBLEM (1000 - 100000 OBJECTS)**
  - \* PROTOTYPING WAYS TO AUTOMATICALLY GENERATE THE TAXONOMY**
  - \* INSERTING THE ASDW TECHNOLOGY INTO AN ONGOING SSFP PROJECT, THE OPAS (OPERATIONS PLANNING AND ANALYSIS SYSTEM) AND PROTOTYPING SUPPORT TO THE STS'S FADS (FLIGHT ANALYSIS AND DESIGN SYSTEM)**
  - \* INITIATED THE PROTOTYPING OF INFORMATION SYSTEM METHODOLOGY TOOLS FOR KNOWLEDGE ACQUISITION AND MODELING**

## **FY89 ACHIEVEMENTS**

All FY89 objectives were successfully met and today, ASDW is a basic shell for supporting the reuse of stored information whether it be about software artifacts or any other design artifact. It contains object management and rule based constraint handling as well as a sophisticated "point and click" textual user interface (called "specification-by-reformulation") that models the way people communicate among themselves. A neural net approach has been implemented to handle the large number of information objects that can be stored and a capability to automatically generate the taxonomy of objects was incorporated. An extensive "Help" system using hypermedia technology was also implemented. The system was put into field testing during the last quarter of FY89. One Beta test supports the OPAS (Operations Planning and Analysis System) project which is part of the SSCC. The other Beta test supports FADS which is associated with the STS's MCC (Mission Control Center).

## **FY89 ACHIEVEMENTS**

- \* ALL EXPECTED RESULTS WERE SUCCESSFULLY ACHIEVED**
- \* SPECIFICATION-BY-REFORMULATION USER INTERFACE**
- \* KNOWLEDGE REPRESENTATION FOR DOMAIN, SOFTWARE, AND "KNOWLEDGE BASE" OBJECTS**
- \* ASSOCIATIVE MEMORY (NEURAL NETS) APPROACH TO HANDLING LARGE NUMBERS OF INFORMATION OBJECTS**
- \* AUTOMATIC TAXONOMY GENERATION OF THE KNOWLEDGE BASE OBJECTS**
- \* HYPERMEDIA "HELP" DOCUMENTATION**
- \* TWO BETA TESTS WERE INITIATED. ONE SUPPORTS THE SSCC AND THE OTHER SUPPORTS THE MCC**

## FY90 OBJECTIVES

The objectives for FY90 are to support the user requested enhancements that develop during field (beta) testing, improve the usability of the system, and expand the scope of the life cycle that is supported. Scale-up must be continually addressed as reuse of a larger portion of the life cycle is addressed. "Internal" scale-up addresses the systems capabilities to efficiently handle large numbers of information objects while "External" scale-up addresses the presentation to the user so that he/she is not overwhelmed by the large amount of stored information. The development of the ESL (Engineering Script Language) will allow non programmers to develop applications and will also support rapid prototyping. The development of a "Management Framework" will be initiated that will guide the users to the correct knowledge acquisition for the system being developed. The development of knowledge acquisition and modeling tools will permit the correct information to be collected. The development of the "integration platform" will produce those utilities and uniform knowledge representations that will integrate these tools and their generated models and knowledge in a way that supports the reuse and sharing of knowledge between models and allows the translation and reuse of knowledge throughout the life cycle phases.

## **FY90 OBJECTIVES**

**\* PROVIDE ENHANCEMENTS BASED ON USER FEEDBACK AS THE ASDW IS PUT INTO FIELD TESTING**

**\* CONTINUE ADDRESSING INTERNAL AND EXTERNAL "SCALE-UP" PROBLEM**

**\* ADD ESL FOR APPLICATION GENERATION AT THE "ENGINEER'S" LEVEL**

**\* PROVIDE A "MANAGEMENT FRAMEWORK" FOR ACQUIRING INFORMATION BASED ON THE CHARACTERISTICS OF THE INFORMATION SYSTEM BEING SUPPORTED**

**\* PROVIDE KNOWLEDGE ACQUISITION PROTOTYPE TOOLS AND AN "INTEGRATION PLATFORM" FOR COLLECTING, REPRESENTING, INTEGRATING, REUSING, AND TRANSLATING CASE KNOWLEDGE THROUGH THE LIFE CYCLE**

## FY90 APPROACH

The ASDW is now moving into phase IV which will significantly broaden its scope of influence. During Phase IV, CASE research and development will continue with the support of four organizations: Inference Corporation, Softech, the KBSL (Knowledge Based Systems Laboratory) at Texas A&M University, and the NRC (National Research Council). Inference will provide the Object management and rule based support for the entire ASDW, the reusable software library management system, and the user interface for accessing parts, for specifying new parts, and for assembling them to create applications. These capabilities will be incorporated into the component called BAUHAUS. Softech will provide an ESL (Engineering Script Language) which should significantly increase productivity in application generation. The ESL is a high level graphical language that permits engineers with a minimum of programming training to design applications that are populated with parts from the BAUHAUS library. Constraint checking of the graph and the parts selection will be provided. The ESL is based upon proven concepts which have been put into operations at the Naval Research Laboratory in a restricted domain. The ASDW effort will demonstrate its utility in the mission planning and analysis domain. The ESL will be integrated with the knowledge based library of reusable parts. The KBSL will add a "Framework" of the information requirements, methodologies (based on extended IDEF methodologies), integrated methodology tools, theory of information modeling (what models are required, when and how to produce them, how to use them, how to integrate their knowledge, etc), and the various developer and user roles across the life cycle. All of the activities in the Framework are configurable to match the characteristics of the information system being supported (eg, business vs engineering). The NRC will provide research into the use of expert systems to support the operational environment of the information system. This study will develop a method to acquire knowledge while an application is being developed, that can be used to help users to easily set up complex input data in the completed application.

## **FY90 APPROACH**

- \* PROVIDE A KNOWLEDGE BASED ENVIRONMENT FOR THE DEVELOPMENT OF SOFTWARE COMPONENTS COMPOSITION SYSTEMS. BUILT BY INFERENCE CORPORATION ON TOP OF ART-IM TO RUN ON MULTIPLE HARDWARE PLATFORMS**
- \* ADD AN ESL THAT IS AN EXTENSION OF THE OPERATIONAL GRAPHICAL LANGUAGE DEVELOPED BY THE NAVAL RESEARCH LABORATORY. THIS PERMITS "END USERS" TO DEVELOP APPLICATIONS**
- \* LEVERAGE USAF SPONSORED ACTIVITIES TO MODEL AND PROVIDE KNOWLEDGE BASED SUPPORT TO THE DEVELOPMENT AND EVOLUTION OF INFORMATION SYSTEMS**
- \* INCORPORATE USAF KNOWLEDGE ACQUISITION AND MANIPULATION CAPABILITIES**
- \* ADD A "MANAGEMENT FRAMEWORK" THAT COORDINATES ALL OF THE ACTIVITIES WITH THE CHARACTERISTICS OF THE INFORMATION SYSTEM**
- \* ADD AN "INTEGRATED PLATFORM" THAT PROVIDES INTEGRATED METHODOLOGIES, UNIFORM KNOWLEDGE REPRESENTATION, AND THE CAPABILITY TO TRANSLATE (REUSE) KNOWLEDGE BETWEEN METHODOLOGIES.**
- \* PROVIDE AN AUTOMATED USERS GUIDE CAPABILITY WHICH WILL BE THE "FRONT END" OF A LARGE, COMPLEX, TRAJECTORY SIMULATIONS SUPPORTED BY ASDW**

## TRANSITION PLANS

Field testing by users in JSC's mission planning community will be a major thrust in FY90. The user interface will be made more graphical with the capability to directly define applications from a block diagram point of view. Expansion of the number of objects will be increased to handle a volume in the neighborhood of 100,000. The modeling, generation, and reuse of early life cycle artifacts will be added along with the capability to use a common representation of their information content and methods for them to share common information. In FY91, a "Management Framework" that will guide the knowledge acquisition and modeling process, the proper use of modeling tools, and the selection of the proper methodologies to be used, as a function of the characteristics of the information system will be developed. This Framework will be supported with a platform of integrated services and a knowledge representation language that will integrate modeling tools used to define the information systems. These modeling tools will enforce the correct use of the methodologies. Also during this period, field testing support will have defined techniques to support users in acquiring knowledge required to operate the developed information system and will provide capability to assist users in the set up of operational data input with knowledge base expert assistance.

All of this CASE research is general in scope and should benefit most NASA programs. Direct use of this research and field testing is currently with the SSCC, the MCC, and the SSE.

## **TRANSITION PLANS**

**\* THE CASE RESEARCH AND DEVELOPMENT IS GENERAL IN ITS SCOPE AND SHOULD BENEFIT MOST NASA PROGRAMS**

**\* SPECIFIC FIELD TESTING IS OCCURRING CURRENTLY IN OPAS (AN SSCC PROJECT) AND FADS (AN MCC PROJECT). BOTH PROJECTS HAVE THE FOLLOWING CHARACTERISTICS THAT ASDW FITS:**

- \* SOFTWARE DESIGN AND CODE REUSE**
- \* INTEGRATION PLATFORMS**
- \* MANAGEMENT OF HUGE AMOUNTS OF INFORMATION**
- \* OPERATIONS HAMPERED BY TOO MANY CONSTRAINTS TO BE EASILY REMEMBERED BY HUMAN USERS**
- \* SHORTAGE OF EXPERIENCED PERSONNEL**
- \* USE AN APPLICATION GENERATOR CONCEPT**

**\* FIELD TEST TEAMS INCLUDE ASDW AND PROJECT PERSONNEL**

**\* SSE NEGOTIATIONS ARE IN PROGRESS**

**\* "FRAMEWORK", INTEGRATION PLATFORM, AND KNOWLEDGE ACQUISITION TOOLS (INCLUDING ENFORCED SYSTEM AND SOFTWARE MODELING METHODOLOGIES) SUPPORT BOTH THE SSE AND OPERATIONAL ENVIRONMENTS SUCH AS THE SSCC**

## CONCLUSIONS

The ASDW project successfully met all of its FY89 objectives and has reached a point where it can be put into useful service. Field testing is underway on two projects that could improve productivity in both the SSCC and the MCC. Productivity is achieved by supporting the reuse of knowledge and software artifacts.

In FY90, user feedback and an increase in the scope of the life cycle supported will be the key drivers. The knowledge base will support both experts and novices. The system will improve the productivity of experts by helping them not to violate constraints that are too numerous for even an expert to constantly stay aware of. Novices will be supported by available knowledge that will support training. Both groups will improve productivity by the guidance in knowledge acquisition processes and the proper selection of tools and methodologies that are provided. Application development will not require as much programmer experience and the operation of applications will be assisted by expertise supplied by the stored knowledge. And finally, the knowledge base will have a uniform representation that permits integration of knowledge across tools, methodologies, and life cycle activities.

## **CONCLUSIONS**

**ASDW IS CURRENTLY A SOFTWARE PARTS COMPOSITION SHELL THAT IS BEING FIELD TESTED FOR PROJECTS IN THE SSCC AND THE MCC. IT SUPPORTS REUSE OF REQUIREMENTS SPECIFICATIONS, DESIGN, SOFTWARE COMPONENTS, AND DATA INPUTS.**

**FY90 AND BEYOND WILL:**

- \* BE DRIVEN STRONGLY BY USER FEEDBACK FOR ENHANCEMENTS**
- \* IMPROVE THE USER INTERFACE**
- \* IMPROVE THE PRODUCTIVITY OF DEVELOPING APPLICATION SOFTWARE BY ALLOWING "END USERS" (EG, NOT PROGRAMMERS) TO DEVELOP APPLICATIONS**
- \* BROADEN THE SCOPE TO COVER MORE OF THE ACQUISITION AND REUSE OF INFORMATION IN EARLY PHASES OF AN INFORMATION SYSTEMS'S LIFE-CYCLE**
- \* DETERMINE THE CORRECT USAGE OF METHODS, TOOLS, PROCESSES, AND KNOWLEDGE AS A FUNCTION OF THE INFORMATION SYSTEM'S CHARACTERISTICS**
- \* PROVIDE KNOWLEDGE OF HOW TO INTEGRATE THE KNOWLEDGE BASES AND METHODOLOGIES REQUIRED FOR INTEGRATED INFORMATION SYSTEMS**
- \* PROVIDE AN AUTOMATED USER GUIDE TECHNOLOGY THAT SHOULD SIGNIFICANTLY REDUCE THE TIME REQUIRED TO GENERATE LARGE APPLICATION INPUT FILES FOR OPAS**
- \* IMPROVE TRAINING AND MINIMIZE THE TIME REQUIRED TO GET NEW PEOPLE PRODUCTIVE EG, HOW TO RUN LARGE APPLICATIONS**

# **Evolution Paths for Advanced Automation**

**Kathleen Healey/EF5  
Kenneth Crouse/EF5**

**Intelligent Systems Branch**

**MITRE Participants: S. Bayer, S. Bell, A. Dorofee, D. Erb,  
C. Kelly, L. Morgan, J. Reynolds, D. Schreckenghost,  
J. Spitzer**

**February 8, 1990**

# Contents

---

- **Goals**
- **Products**
- **Recommendations and Results**

## Goals

---

- To enable, promote, and accelerate appropriate use of advanced automation on Space Station Freedom by
  - identifying existing paths, recommending improvements to existing paths, and recommending new paths for incorporating advanced automation technology into the SSFP
  - generating specific plans for creating and using these paths to incorporate advanced automation into software facilities within the SSFP, specifically the test beds and the Software Support Environment (SSE)

## **Products**

---

- **Space Station Freedom Program Advanced Automation:**
  - **Volume I - Evolution Paths**
  - **Volume II - Evolution Within Test Beds**
    - **Series 1 - A Generic Plan for SSFP Test Beds**
    - **Series 2 - Integrated Test Beds: Lessons Learned from the Data Management System Test Bed**
  - **Volume III - Evolution With Environments**
    - **Series 1 - Concepts for a Software Prototype Transition Environment (SPTE)**
    - **Series 2 - A Plan for the Software Support Environment**

## **Products (cont.)**

---

- *Space Station Freedom Program Capabilities for the Development and Application of Advanced Automation*
- All documents will be published in February 1990. Preliminary copies were distributed to the Advanced Technology Advisory Committee for review prior to that committee's February meeting.

## **Volume I: Evolution Paths Recommendations**

---

- Establish a coordinating organization to direct and manage the diffusion of advanced automation from innovators into the programs
- Establish Software Prototype Transition Environment (SPTE) to provide near-operational, near-real-time environments with a super-set of SSE software development tools
- Other detailed recommendations were made per a model for advanced automation diffusion, such as identifying standards and methods, improving and expanding communications, improving selection of advanced automation projects, and identifying paths for technology transfer.

## **Volume II, Series 1: A Plan for the Test Beds Recommendations**

---

- Each test bed should develop a specific evolution plan prior to incorporation of advanced automation
- Projects in design and prototyping test beds should focus on fieldable applications based on user needs
- Projects in research test beds should focus on applied research driven by application needs
- Communication among SSFP projects should be established for the distribution of lessons learned and promising new technologies and tools

# **Volume II, Series 2: Integrated Test Beds: Lessons Learned from the DMS Test Bed Recommendations**

---

- **Hold test bed integration meetings with full participation for information exchange between projects**
- **Improve test bed visibility through definition of specific products, reviews, reports and demonstrations, and distribution of products**
- **Use improved visibility to expand participation, increase funding, and extend benefits of integration efforts to other areas**
- **Develop strategic hardware evolution plans**
- **Identify existing or establish new communication paths for transfer of lessons learned on test beds**

# **Volume III, Series 1: Concepts for a Software Prototyping Transition Environment Concepts**

---

- **Fills the gap between relatively unconstrained test beds and the constrained SSE and operational environments for more effective advanced automation diffusion**
- **Supports engineering and incorporation of advanced automation prototypes, tools, hardware, and re-engineered operational software**
- **Supports a consortium of industry and government partners working as a team**
- **Allows SSFP contractors to work closely with users in developing prototypes for operational environments**
- **Supports incremental software development using prototypes**

# **Volume III, Series 1: Concepts for a Software Prototyping Transition Environment Recommendations**

---

- The SPTE can be implemented by interconnecting existing facilities with high level services layer
  - Low-impact, low-cost solution
  - Allows existing test beds, labs, and SSE facilities to participate together
    - e.g., MOSL, TFCR, MPAC test bed, and JSEL
- Use COTS software to perform the integration, e.g., Cronus (BBN Systems and Technologies)

## **Volume III, Series 2: A Plan for the SSE Recommendations**

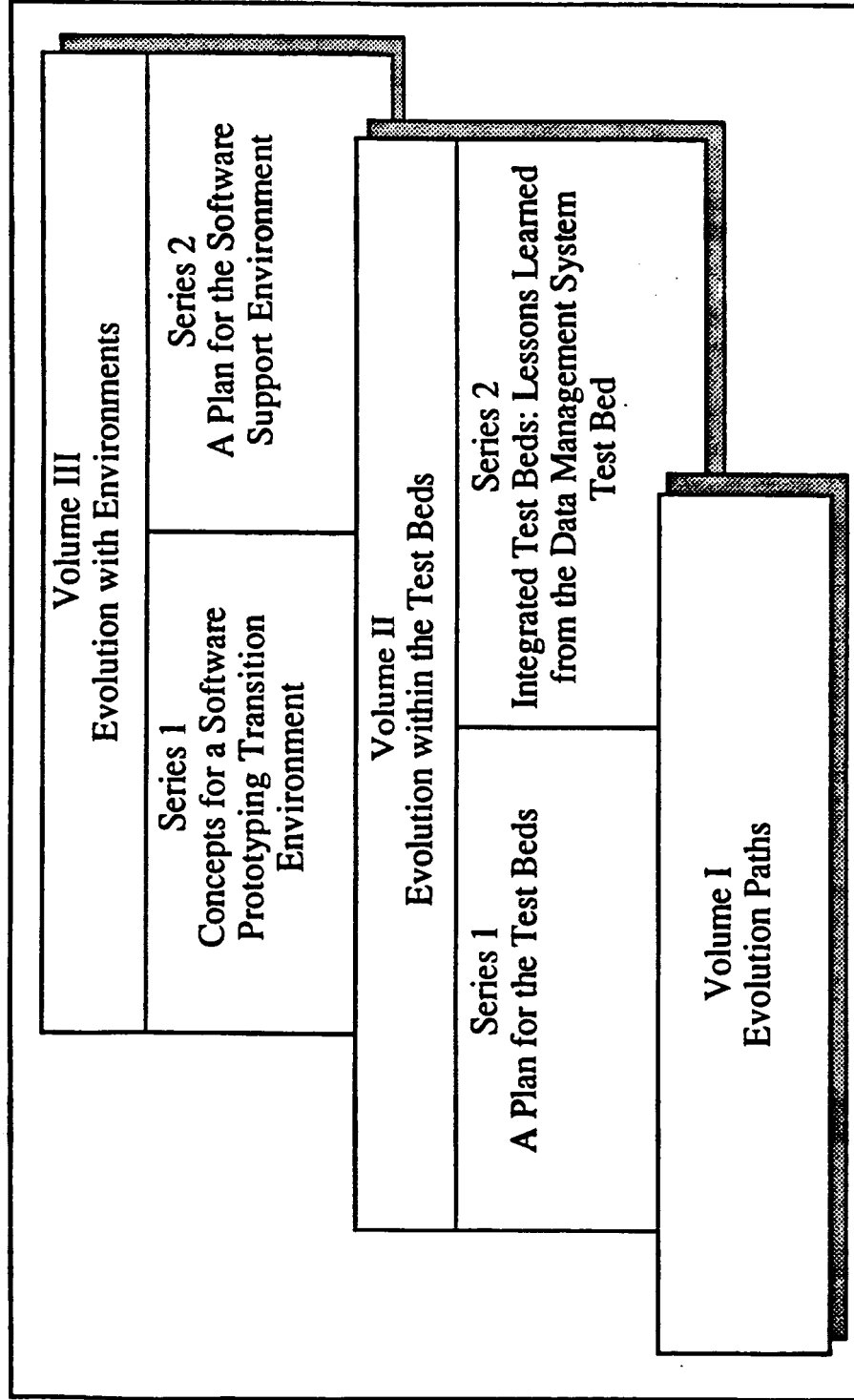
---

- **Establish and maintain organizational components, methodologies, and activities to enhance the incorporation of advanced automation technology into the SSE**
- **Continue definition and development of plans and procedures for the management of advanced automation diffusion, particularly DRLI 14, Plan for Implementing Non-SSE Generated Software, to allow users to bring in applications developed with newer technology than what the SSE currently supports**

# **Space Station Freedom Program Capabilities for the Development and Application of Advanced Automation**

---

- **Contents of Space Station Freedom Program  
Capabilities for the Development and Application of  
Advanced Automation:**
  - Reference document identifying capabilities  
and plans for advanced automation within the  
SSFP
  - Includes test beds and research laboratories at  
various NASA facilities



**Volumes in the Space Station Freedom Program  
Advanced Automation Series**

**SESSION IX**  
**ADVANCED AUTOMATION ENVIRONMENTS**

**Session Chair:**  
**Dr. Henry Lum**  
**NASA Ames Research Center**



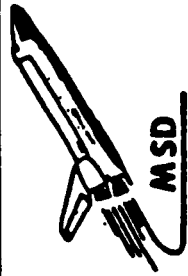
MISSION SUPPORT DIRECTORATE

JSC

# ART/Ada and CLIPS/Ada

Chris Culbert  
Project Manager  
NASA/Johnson Space Center  
Software Technology Branch/FR5  
Houston, TX 77058

FY89 Annual Report



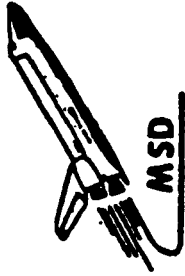
## Introduction

### Background

- Expert systems have demonstrated applicability to NASA domains and problems.
- Original development environments not very useful for Space Station Freedom.
  - SSFP will need deeply integrated, embedded applications.
- Ability to apply automation could be enhanced by use of Ada based expert system tools.
  - Few available products to consider.

### Goals

- Conduct research into the implementation in Ada of state-of-the-art expert system tools.
- Develop and evaluate incremental prototypes of Ada based expert system development tools.
  - Use commercial product ART-IM as one starting point. Expand to full functionality tool.
  - Use NASA based CLIPS as a second starting point. Less functionality than ART, but ready for use much sooner.
- Provide Ada based tools to Space Station Freedom applications for evaluation and benchmarking.



**NASA**

MISSION SUPPORT DIRECTORATE

**JSC**

**MSD**

## **FY89 Objectives and Approach - ART/Ada**

Four discrete tasks planned for FY89:

- Review and analyze Phase I prototype (kernel run-time inference engine). Develop plans and designs for extensions to allow following capabilities:
  - a schema knowledge representation language
  - a justification-based truth maintenance system
  - explanation support utilities
- Develop and refine algorithms required to implement new capabilities.
- Implement in Ada algorithms and procedures for new capabilities. Produce functional prototypes with extended capabilities.
- Conduct performance analysis of prototype(s).



## **FY89 Objectives and Approach - CLIPS/Ada**

Three primary tasks for FY89:

- Complete major redesign to improve performance and reduce size.
  - Alpha version based upon code translation. Did not make effective use of Ada.
  - Redesign based upon object oriented approach to improve code modularity.
  - Maintain full syntactic compatability with C version of CLIPS.
- Add new functionality to CLIPS/Ada to make consistent with current C release.
  - Alpha version based upon CLIPS 4.1 release.
  - Current CLIPS release is version 4.3
- Thoroughly test CLIPS/Ada to eliminate bugs and mistakes common to first release software.



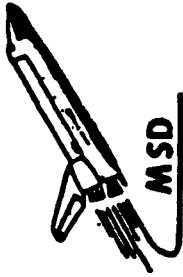
## **FY89 Accomplishments - ART/Ada**

- Recoding in Ada of run-time kernel for the ART-IM inference engine was completed and demonstrated in February 1989 (completion of Phase I activities).
- Extensions and plans to implement new features were completed (Task 1).
- Approaches to implementing new features were identified and tested. (Task 2).
- Coding was begun to provide two more prototypes.
  - First prototype will include most ART-IM capabilities except schema and viewpoints. Due to be completed by early FY90 (November timeframe).
  - Second prototype will include schema system. Due to be delivered in February 1990.
- Plans for extended evaluation of ART/Ada prototypes completed and beta sites lined up.
  - Multiple Air Force and NASA sites selected for FY90 testing.



## **FY89 Accomplishments - CLIPS/Ada**

- First release version of CLIPS/Ada completed for workstations.
  - Nearly 100% syntax compatability with C version of CLIPS 4.3
  - Performance significantly improved; now runs only 2 to 3 times slower than C version.
- Extensive testing performed by independent CLIPS test team.
  - Numerous errors detected and corrected.
  - PC version undergoing testing at present.
- Plans completed for further extensions and performance enhancements for CLIPS/Ada.



**NASA**

MISSION SUPPORT DIRECTORATE

**JSC**

**MSD**

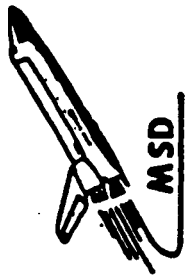
## **FY90 Plans**

### **ART/Ada Plans**

- Extended evaluation of both ART/Ada prototypes for most of FY90.
  - First prototype should be available for use in early FY90.
  - Second prototype to be sent to test sites in February 1990.
  - Final evaluations from all sites due to JSC by July 1990.
- Continued development to improve ART/Ada for use in real-time, distributed applications.

### **CLIPS/Ada Plans**

- Complete PC version of CLIPS/Ada (if compilers are capable enough)
- Further improve performance by redesigning elements of the pattern matching system.
- Add extensions currently under development for next release of C version.
- Fix bugs as they are identified in the release version of CLIPS/Ada 4.3
- Examine enhancements to support object description capability in CLIPS/Ada and C version of CLIPS.



## Conclusions

- Considerable progress has been made towards our goal of demonstrating the feasibility of implementing expert system tools in Ada.
- Groundwork laid towards providing this technology to the Space Station Freedom Program.
  - CLIPS accepted into the SSE.
  - CLIPS/Ada delivered to the SSF world for general use.
  - ART/Ada prototype to be used in SSFP prototype development in multiple work packages.
- Clear, consistent upgrade path for the use of expert systems in the SSFP has been identified and elements put in place.
- Remaining issues deal with improving performance and examining use in key applications such as real-time and/or distributed systems.

# **KNOWLEDGE-BASED SYSTEMS VERIFICATION AND VALIDATION**

**Sally C. Johnson**  
**System Validation Methods Branch**  
**Information Systems Division**  
**Langley Research Center**

**FTS: 928-6204**  
**NASAMail: SCJOHNSON**

**PHASE I - TO BE COMPLETED BY AUGUST 1990**

**TECHNIQUES FOR SOFTWARE ENGINEERING AND ANALYSIS OF RULE-BASED SYSTEMS**

**GUIDELINES FOR DEVELOPING FAULT DETECTION, ISOLATION, AND RECOVERY KBS'S**

**PHASE II - NOT FUNDED**

**EVALUATE TECHNIQUES FOR OTHER KNOWLEDGE REPRESENTATIONS**

- o MODEL-BASED SYSTEMS**
- o FRAME-BASED SYSTEMS**
- o BLACKBOARD ARCHITECTURES**

**EXTEND GUIDELINES TO COVER OTHER KBS APPLICATIONS**

- o CONTROL AND MONITORING**
- o TRAINING SYSTEMS**
- o PLANNING AND RESOURCE MANAGEMENT**

# **PHASE I - GUIDELINES FOR RULE-BASED SYSTEMS FOR FDIR APPLICATIONS**

## **APPROACH**

### **ASSESSMENT OF STATE-OF-PRACTICE**

#### **ASSESSMENT OF TECHNIQUES**

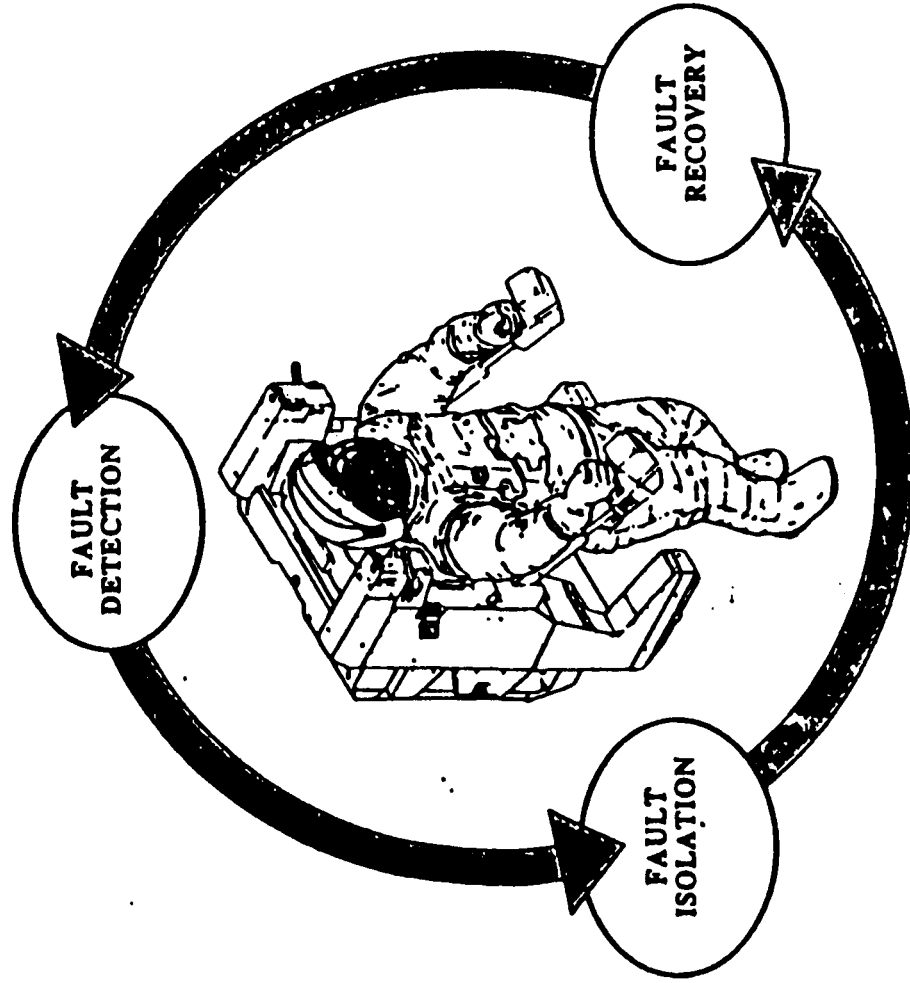
- EVIDENCE-FLOW GRAPH TECHNIQUES
- SPECIFICATION AND KNOWLEDGE REPRESENTATION STRATEGIES
- MODULARIZATION TECHNIQUES FOR RULE-BASED SYSTEMS

#### **RECOMMENDATION OF KBS V&V REQUIREMENTS**

#### **RECOMMENDATION OF MODIFICATIONS TO SOFTWARE SUPPORT ENVIRONMENT**

## PHASE I EXAMPLE APPLICATION

### FAULT DETECTION ISOLATION AND RECOVERY (FDIR) FOR THE MANNED MANEUVERING UNIT (MMU)



- REPRESENTATIVE OF FDIR SYSTEMS
- COMPLEX SPACE APPLICATION
- PUBLICLY AVAILABLE
- IMPLEMENTED IN CLIPS 4.1
- 104 RULES AND 45 FACTS

## **PRELIMINARY RESULTS**

**KNOWLEDGE BASE OF MMU FDIR EXAMPLE IS DIFFICULT TO COMPREHEND**

- **ENUMERATION OF POSSIBLE CASES FROM A DECISION TABLE**
- **FLAT RULE BASE, NO HIERARCHY OR MODULARITY**
- **LIMITED FDIR CAPABILITY**

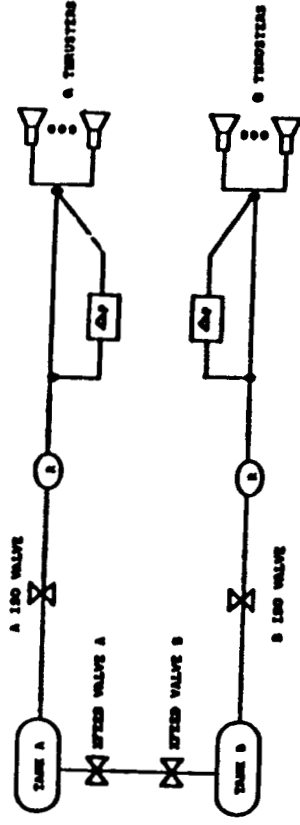
## **FOCUS OF RESEARCH**

- **SRI - DEVELOPMENT METHODS FOR FDIR SYSTEMS**
- **INHOUSE/VIGYAN - MODULARIZATION SCHEMES**
- **WPI - ANALYSIS AND TESTING TECHNIQUES**

**DEVELOPMENT METHODS FOR FDIR SYSTEMS**

**SRI INTERNATIONAL**

## DEVELOPMENT OF FDIR SYSTEMS



## MODEL OF SYSTEM OPERATION

### KNOWLEDGE REPRESENTATIONS FOR ENCODING MODEL

- MORE UNDERSTANDABLE
- MORE COMPREHENSIVE FDIR
- MORE EASILY MAINTAINABLE
- LESS ERROR-PRONE DEVELOPMENT

**RULE GROUPINGS FOR  
RULE-BASED EXPERT SYSTEMS**

**INHOUSE/VIGYAN INC.**

**GOAL -- DEVELOPMENT OF TOOLS AND STRATEGIES FOR  
USER-DIRECTED PARTITIONING OF CLIPS RULE BASES**

**WHY?**

- o AID IN COMPREHENSION - VIEW SYSTEM FROM HIGHER SEMANTIC LEVEL
- o PROVIDE "FIREWALLS" AROUND CRITICAL PROBLEM SUBDOMAINS



**RELIABILITY AND MAINTAINABILITY**

**HOW?**

- o INVESTIGATED VARIOUS GROUPING STRATEGIES --
  - LINDELL; JACOB AND FROSCHER; LINDENMEYER; ETC.
- o OBTAINED RULE GROUPING SOFTWARE FROM JACOB
- o DEVELOPED TRANSLATOR FROM CLIPS RULES TO JACOB'S RULE FORMAT
- o INVESTIGATING METHODS FOR REPRESENTING RULE CLUSTERING INFO



**RECOMMEND RULE GROUPING STRATEGIES FOR VARIOUS TYPES OF KBS'S**

## MODULARIZATION OF RULE BASES

### RULES

```
(RULE engine-and-radar
  (IF (engine-status = ready)
    (radar-status = ready))
  (THEN (ship-status = ready)))

(RULE gas-and-oil
  (IF (engine-fuel-level >= 100)
    (engine-lubricated = true))
  (THEN (engine-status = ready)))

(RULE ... etc.))
```

### GROUPED RULES

```
(GROUP ship-ready
  (PRODUCE ship-status "ready means ship can be sailed on any mission
    up to 1000 nautical miles")
  (USE engine-lubricated))

(RULE engine-and-radar
  (IF (engine-status = ready)
    (radar-status = ready))
  (THEN (ship-status = ready)))

(GROUP engine-ready
  (PRODUCE engine-status "ready means engine fully operational")
  (USE engine-lubricated))

(RULE gas-and-oil
  (IF (engine-fuel-level >= 100)
    (engine-lubricated = true))
  (THEN (engine-status = ready)))

(RULE ... etc.))

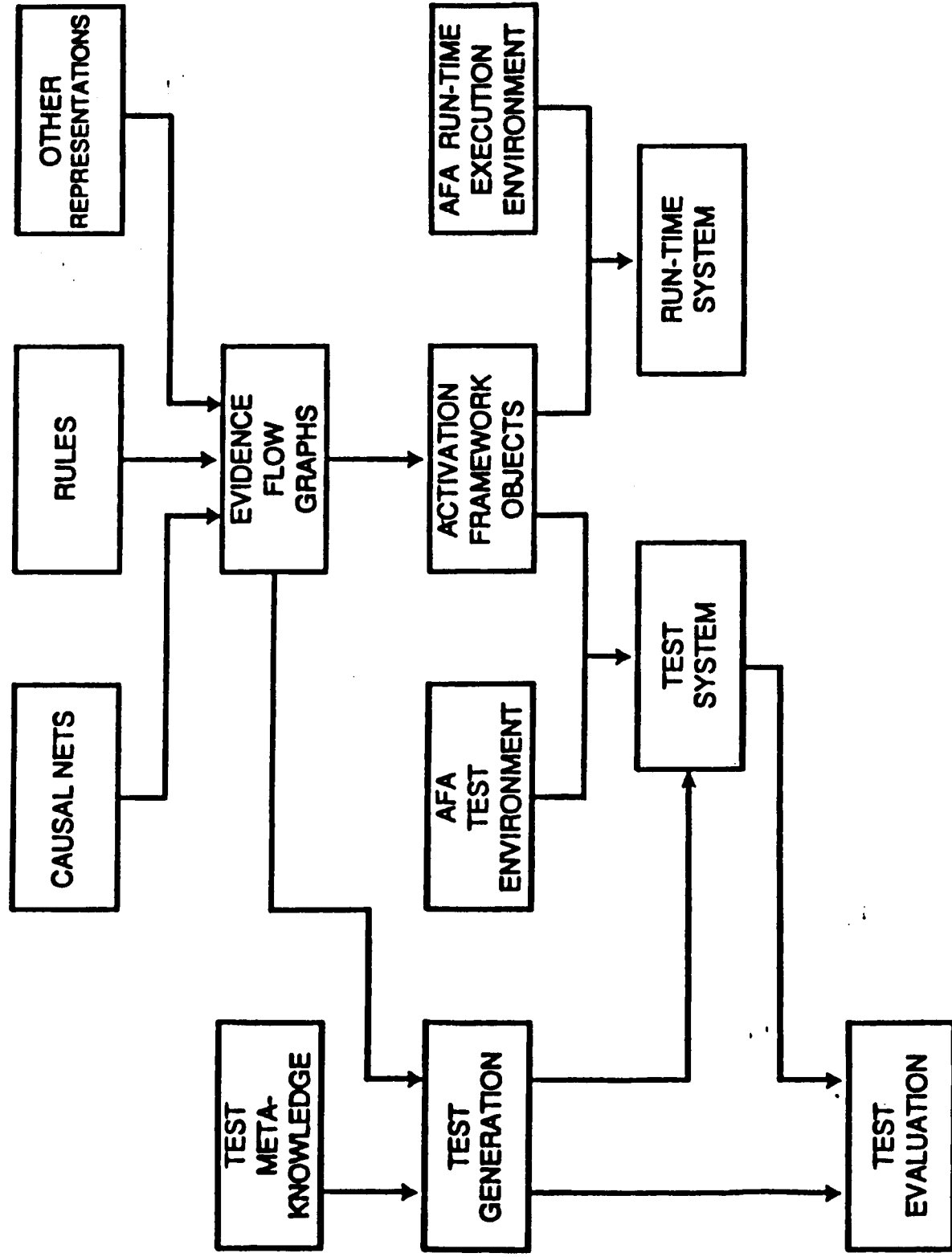
(GROUP radar-ready
  (PRODUCE radar-status "ready means all radars fully
    operational")
  (USE engine-lubricated))

(RULE ... etc.))
```

**EVIDENCE FLOW GRAPH  
ANALYSIS AND TESTING TECHNIQUES**

**WORCESTER POLYTECHNIC INSTITUTE**

# CONVERSION OF RULES INTO ADA CODE



# VALIDATION AND VERIFICATION

- LOOKING FOR ERRORS
  - DOES NOT MEET SPECIFICATIONS
  - PRODUCES FATAL OUTPUT
  - INCONSISTENT AND UNSTABLE OUTPUTS
  - OUTPUTS SENSITIVE TO
    - DATA INPUT ORDER
    - EXECUTION ORDER
    - SMALL CHANGES IN INPUT VALUE
    - SMALL CHANGES IN PARAMETERS
  - POTENTIAL FOR DEADLOCK, LIVELOCK, INFINITE LOOPS
- METHODS
  - GRAPH ANALYSIS - SOME IS DONE DURING TRANSLATION TO FLOW GRAPH
  - TEST SIMULATION DONE USING TEST VERSION OF ADA
  - PETRI NET ANALYSIS

# **TEST LANGUAGE DEVELOPMENT**

**PIONEERING RESEARCH TO DEVELOP A FORMAL  
PERFORMANCE SPECIFICATION LANGUAGE**

**SYNTAX DEFINES**

**POSSIBLE INPUTS**

**LIKELY INPUT VALUES**

**MUST HAPPEN TEST CASES**

**MUST NOT HAPPEN CASES**

**ORDERING CONSTRAINTS**

**VALUE CONSTRAINTS**

**USED FOR TEST GENERATION AND EVALUATION**

# Test Configuration Metaknowledge

Specifies how tests are to be generated

Types of tests within test space

- fully random

- high end of normal range

- average value in normal range

- hit every part of the distribution  
etc.

Number of tests to run

- number based on a confidence

- user input number

Timing of inputs

- Average time between inputs (distribution etc.)

- Run a set until system settles down  
etc.

## **CONCLUSIONS**

**EVALUATIONS OF THREE TECHNIQUES WILL BE COMPLETED SPRING 1990**

**PRELIMINARY SET OF GUIDELINES WILL BE COMPLETED AUGUST 1990**

- o SOFTWARE ENGINEERING AND ANALYSIS OF RULE-BASED SYSTEMS**
- o SPECIFIC TO MONITORING AND DIAGNOSIS APPLICATIONS**

**EXTENSION OF THESE GUIDELINES IS NEEDED**

- o OTHER KNOWLEDGE REPRESENTATIONS**
- o OTHER TYPES OF KNOWLEDGE-BASED SYSTEMS**

**The  
Software Support Environment  
Design Knowledge Capture  
(SSE/DKC)  
Project**

**Tom Dollman  
Information & Electronic Systems Laboratory  
Marshall Space Flight Center**

**February 8, 1989**

## **Overview**

- o **Background**
- o **Objective**
- o **Approach**
- o **HSTDEK/HSTORESIS Description**
- o **Motivation for DKC**
- o **SSE Description**
- o **Software Evaluations**
- o **Recommendations**
- o **Plans**
- o **Participants**
- o **Summary**



## **Background**

- o **The Software Support Environment (SSE)**
  - is a multi-facility environment
  - supports the life-cycle management of all operational software
- o **Design Knowledge Capture (DKC) requirement/approach in "Process Requirements Document for Design Knowledge Capture".**
- o **DKC is a broad engineering activity aimed at collecting & retaining (in electronic form) the design & engineering expertise utilized on the design process.**
- o **DKC products will be supporting numerous ground & flight applications**
- o **Current SSE:**
  - Ada development environment
  - "critical to evaluate the SSE as a mechanism to capture system design knowledge"
- o **This task leverages off of other DKC technology development**
  - OAST
  - Other OSS

## **Background**

- o The Software Support Environment (SSE) is the multi-facility environment which will support the life-cycle management of all Space Station Freedom Program (SSFP) operational software, including both flight & ground software.
- o The SSFP has established a requirement for a Design Knowledge Capture (DKC) effort integral to the development of the Space Station Freedom, and has documented an approach in "Process Requirements Document for Design Knowledge Capture".
- o DKC is a broad engineering activity aimed at collecting & retaining (in electronic form) the design & engineering expertise utilized on the design process.
- o The overall DKC activity will result in a set of knowledge bases and knowledge-based systems supporting numerous ground & flight applications in Space Station Freedom. These software elements should be categorized as operational S/W, hence be in SSE.
- o Currently, the SSE is an Ada development environment, with little provision for accommodating knowledge-based systems (cf., "Space Station Advanced Automation Study Final Report" - it is "critical to evaluate the SSE as a mechanism to capture system design knowledge".) This SSE/DKC task is aid in this evaluation.
- o This task leverages off of other DKC technology development work being funded by the Office of Aeronautics and Space Technology and the Office of Space Station



SSE/DKC

## Objective

The objective of this task is to assess the potential for using the Space Station Freedom Program's (SSFP's) Software Support Environment (SSE) workstations and associated software for DKC tasks.

This assessment will include the identification of required capabilities for DKC and hardware and software modifications needed to support DKC.

## **Approach**

**Approaches to achieving this objective are as follows:**

- **Research the problem of knowledge engineering in a Computer-Aided Software Engineering (CASE) environment;**
- **Research the problem of applying SSE CASE tools to develop knowledge based systems; and**
- **Directly utilize SSE workstations to support a DKC activity.**

**Initial year (FY89) focus:**

- **evaluate the capability of the SSE System to support DKC activities**
- **make empirical observations and develop critical analyses**
- **identify specific enhancements to the SSE System**
- **utilize (non-toy) knowledge based products to evaluate software development tools on the SSE workstation**
- **utilize ongoing knowledge based project to evaluate management tools and methodologies on the SSE host**

## **Approach**

**Approaches to achieving this objective are as follows:**

- Research the problem of knowledge engineering in a Computer-Aided Software Engineering (CASE) environment;**
- Research the problem of applying SSE CASE tools to develop knowledge based systems; and**
- Directly utilize SSE workstations to support a DKC activity.**

**Initial year (FY89) focus was to evaluate the capability of the SSE System to support development and management of knowledge based systems developed for DKC activities. The products of the evaluation are empirical observations and critical analyses necessary to identify specific enhancements to the SSE System to improve its compatibility with knowledge based system development.**

**Knowledge based products developed in HSTDEK provide test cases for evaluating software development tools on the SSE workstation.**

**The management approach embodied in the HSTDEK project provides a test case for evaluating the management tools and methodologies offered on the SSE host computer.**



SSE/DKC

## HSTDEK/HSTORESIS Description

- o Hubble Space Telescope Design Engineering Knowledgebase (HSTDEK) Project:
- Sponsored by OAST - a CSTI element since FY88
- o Primary goal is to enable major NASA projects to capture the design & engineering expertise that accumulate during the development of their systems. Use HST as "testbed"
- o Acquire knowledge from multiple domain experts representing different technical disciplines, then integrate. Use The Hubble Space Telescope Safemode Investigation System (HSTORESIS) to test methods
- o HSTORESIS will demonstrate the application of expert design knowledge to vehicle anomaly (safemode) analysis during the orbital verification phase of the HST.
- o Key features of HSTORESIS that the SSE should be able to accommodate are:
  - Object oriented programming
  - Knowledge representation
  - Rule base
  - Rapid prototyping

## **HSTDEK/HSTORESIS Description**

- o Hubble Space Telescope Design Engineering Knowledgebase (HSTDEK) Project:
  - Sponsored by Office of Aeronautics and Space Technology under the Civilian Space Technology Initiative - Initiated FY88
- o The primary goal of HSTDEK is to enable major NASA projects to capture the design & engineering expertise that accumulate during the development of their systems. The HST is being used as a suitable environment within which this new technology can be developed and validated.
- o One element of this goal is to develop a methodology for acquiring knowledge from multiple domain experts representing different technical disciplines, and integrating it into a single knowledge base. Suitable methodologies are being investigated via the development of a demonstration - The Hubble Space Telescope Safemode Investigation System (HSTORESIS)
- o HSTORESIS will demonstrate the application of expert design knowledge to anomaly (safemode) analysis during the orbital verification phase of the HST.
- o Key features of HSTORESIS that the SSE should be able to accommodate are:
  - Object oriented programming (KEE units)
  - Knowledge representation (frames)
  - Rule base (for representing, in this case, heuristic knowledge)
  - Rapid prototyping (using a Boehm spiral model)



6873

## Motivation for Design Knowledge Capture (DKC)

- o System design knowledge currently:
  - factual knowledge in (paper or electronic) documents
  - not much experiential knowledge is saved, may not be recoverable later
- o "the process of capturing, analyzing, and maintaining design knowledge in a machine-interpretable form ... to formalize and retain as much of the design and engineering expertise as will be required to support the effective utilization of knowledge based systems to increase system efficiency, productivity, and reliability during operations."
- o Technology issues facing DKC are knowledge acquisition and representation. Separate issues, but related since:
  - the representation chosen may affect the acquisition process,
  - the acquisition process can suggest useful representations, and
  - some knowledge should stay in the form in which it is available (e.g., text files representing data).

Thus, SSFP needs flexible knowledge base development environment(s)

## Motivation for Design Knowledge Capture (DKC)

- o Currently knowledge about the design of a system/subsystem is primarily factual knowledge that is captured in paper (or electronic) documents during the early phases of the life-cycle. These documents frequently fail to capture the experiential knowledge applied by experts when making critical design decisions.
- o Experiential knowledge may not be recoverable from the expert, later
- o Within the context of the SSFP, DKC is defined as "the process of capturing, analyzing, and maintaining design knowledge in a machine-interpretable form." The goal of DKC for the SSFP is "to formalize and retain as much of the design and engineering expertise as will be required to support the effective utilization of knowledge based systems to increase system efficiency, productivity, and reliability during operations."
- o Technology issues facing DKC are the acquisition of design knowledge and the representation of knowledge. Separate issues, but related since:
  - the representation chosen may affect the acquisition process,
  - the acquisition process can suggest useful representations, and
  - it is possible that some of the knowledge a system is to use should stay in the form in which it is available (e.g., text files representing data).

Thus, a knowledge base development environment which offers flexibility becomes essential for large projects such as the SSFP.

## **SSE Description**

- o The SSE System includes hosts and workstations, systems software, and comm. networks
- o The SSE (part of the SSE System)
  - consists of "software, procedures, standards, hardware specifications, documentation, policy and training material"
  - collectively provide the "environment to be used for the life-cycle management" of all operational software for the SSFP.
- o The SSE version for this evaluation is Operational Increment 2 SSE System, includes:
  - COTS & proprietary software for application software development on VAX & IBM 3090;
  - direct access to on-line training modules from the user's workstation.
- o SSE evaluation focus
  - MacIntosh II SSE Workstation (augmented with TI microExplorer hardware/software and KEE)
  - SSE Development Facility VAX host at Johnson Space Center.

## **SSE Description**

**The SSE System "is a multi-facility environment consisting of computer hardware, including hosts and workstations, systems software, communications networks, and the SSE that fully supports the life-cycle management of the SSFP operational software... and the SSE System itself"**

**The SSE, part of the SSE System, consists of "software, procedures, standards, hardware specifications, documentation, policy and training material" which collectively provide the "environment to be used for the life-cycle management" of all operational software for the SSFP.**

**The SSE is being developed in 4 stages over a six-year period. The version for this evaluation is Operational Increment 2 (OI-2) (initial) SSE System. OI-2 includes Commercial off-the-shelf software & proprietary software to permit software development on a DEC VAX system, and an IBM 3090 computer which provided the ability to access computer-based training modules directly from the user's workstation.**

**This elements of the SSE System environment in which we evaluated the SSE Workstation consisted of the Macintosh II SSE Workstation (augmented with additional hardware & software) and the SSE Development Facility VAX host at Johnson Space Center.**

**The hardware elements of the Macintosh II SSE workstation used for this evaluation were: 5MB Apple memory extension, three 80MB Winchester hard disk drives, one 800KB 3.5 inch microfloppy diskette drive, one 13-inch 640-by-480 pixel high-resolution color monitor, NuBus video expansion card, 265 KB video controller memory, mouse, Apple extended keyboard, one external Everex EMAC-MD2400 Modem. Augmentations to this configuration were a Texas Instruments microExplorer processor board, and 8 MB Texas Instruments memory extension.**

## **Workstation Software Evaluation**

- o **SSE Workstation provides software development environment**
- o **Evaluate CASE tools on SSE workstation. Focus on support for design capture activities of HSTORESIS and tools integration. Develop tests to:**
  - **represent the communication between a satellite telemetry monitor object, the HSTORESIS data management module, and an HST device model;**
  - **test for Iconix PowerTools inconsistencies;**
  - **evaluate abstract data type representation/reuse**
- o **Knowledge Engineering Environment (KEE)**
  - **commercial Lisp-based shell for developing knowledge-based systems (not in SSE Baseline)**
  - **effectively integrates support for both rule-based and object-oriented systems**
  - **static displays easy to build, but high-quality, dynamic displays more difficult**
  - **KEE port to microExplorer effective**
- o **Distributed APCE macros/applications programs**
  - **not supported via modem, not evaluated**

## Workstation Software Evaluation

- o The SSE workstation provides a software development environment
- o Evaluate Computer Aided Software Engineering (CASE) tools delivered with the subject SSE workstation. Focus on how well the tools support the design capture activities of HSTORESIS and how well the tools are integrated with one another (particularly important due to need for requirements traceability). Approach was to develop tests to:
  - represent the communication between a satellite telemetry monitor object, the HSTORESIS data management module, and an HST device model;
  - check whether or not the PowerTools data dictionary facility was able to detect multiple entries under the same identifier name;
  - try and represent an abstract data type (encapsulated as an object) and then reuse it in another specification (utilizing John Guttag's method for the algebraic specification of abstract data types to define axioms for a stack object - the stack axioms were then used to generate the program requirements entered into other FreeFlow tool).
- o KEE (not in SSE Baseline) is a frame-based system which effectively integrates support for both rule-based and object-oriented systems. The tools for building visual interfaces make it very easy to build static displays (useful for debugging & prototyping), but building a high-quality, dynamic user interface is considerably more difficult. The KEE port to microExplorer appears to be very successful, although the standard Macintosh screen is too small to be effective.
- o Neither the Iconix CASE tools nor the APCE (Automated Program Control Environment) accommodates rules

## Workstation (Iconix PowerTools) Software Evaluation

- o FreeFlow
  - Data Flow Diagrams (DFD's) are built, minispecs are written, and PDL source code is generated
  - no graphical distinction between a discrete & continuous data flow
  - no method for representing a control transformation.
  - one-way only from minispecs to PDL Data Dictionary
- o PowerPDL
  - PDL source is converted to PDL listing file & a structure chart
  - create PDL listing files from PDL source files
  - extracts PDL statements from some source-program comments, no consistency check
- o SmartChart uses structure chart to provide graphical representation of system
- o Overall, no support for rules

## **Workstation (Iconix PowerTools) Software Evaluation**

- o FreeFlow provides a convenient method for doing standard data flow diagrams, but with no graphical distinction between a discrete & continuous data flow, nor a method for representing a control transformation. Also, the generation of PDL from the minispecs is a one-way process. (No way to move PDL-file changes back up to Data Flow Diagram.)
- o PowerPDL's main function is to create PDL listing files (and a structure chart) from FreeFlow's PDL source files. Also can extract PDL statements embedded as comments in source code (did not work with Lisp). No consistency enforcement, the coder bears full responsibility for having "correct comments".
- o SmartChart uses structure to provide graphical representation of system

## **Host Software Evaluation**

**SSE Development Facility (VAX) provides software project management:**

- o **Project management tools were to have been distributed between the workstation and the SSEDV VAX. Some were unavailable at the time; also it was difficult to adapt the Iconix tools to HSTORESIS design requirements**
- o **The knowledge engineering (KE) model used is an object-oriented, Lisp-based environment in which heuristic knowledge is encoded in IF-THEN rules.**
- o **Function Point Analysis (FPA)**
  - **used to estimate count of software lines of code and lines of documentation**
  - **Inadequate due to problems in:  
defining a (Lisp) line of code  
determining the number of rules required to answer a particular question; and  
developing a functional analog for a rule that is applicable to FPA.**
- o **Cost Construction Model (COCOMO) is used to estimate software & manpower costs**
  - **accuracy of COCOMO depends on the output of the FPA tool**
  - **rules and rule complexity have not been incorporated into the COCOMO cost drivers.**

## Host Software Evaluation

The SSE Development Facility (SSEDF) VAX provides an environment to manage software projects.

- o Evaluation includes project management tools distributed between the workstation and the SSEDF VAX. These tools were not tested as thoroughly as the SSE software development tools, due in part to their physical unavailability at the time, and the difficulty in adapting the Iconix tools to the design requirements of HSTORESIS. The methodologies of each system was studied so as to be evaluated with respect to knowledge engineering (KE) activities.
- o The KE model used is an object-oriented, Lisp-based environment in which heuristic knowledge is encoded in IF-THEN rules.
- o Estimation of code & document sizes are supported by Function Point Analysis (FPA). FPA is inadequate for this KE model due to problems in: defining a (Lisp) line of code - which can be a number of readable lines; determining the number of rules required to answer a particular question; and developing a functional analog for a rule that is applicable to FPA.
- o COCOMO (Cost Construction Model) is available to estimate development time & manpower requirements for the development of the products. The accuracy of COCOMO depends on the output of the FPA tool - and is of questionable value with this KE model. Furthermore, rules and rule complexity have not been incorporated in the COCOMO cost drivers.

## Host Software Evaluation

- o Selected KBS life-cycle is (*process-driven*) Boehm spiral development model.
- o ARTEMIS (Automated Reporting, Tracking, and Evaluation Management Information System)
  - enter project plan, then use for planning schedules & tracking their progress
  - appears capable of managing the schedules of a KBS development project.
- o APCE (Automated Program Control Environment) provides a method to create and control a project instance:
  - user provides project plan and schedule (e. g., from ARTEMIS)
  - is an extensive tool-set to enable access to project database. (Database is project-specific database for all defined products, schedules, and other project information.)
  - project instance creation adequate
  - provided the creator with ability to tailor the project development phases (w/ maximum of five phases)
  - *event-driven* control mechanism
  - Linear list of discrete events that require some formal notification in order to transition from one phase (e.g., requirement) to another (e.g., development); backup to previous phase is anomaly
  - no support for rules

## Host Software Evaluation

- o A project instance was created modelled on (Barry Boehm's) spiral development cycle, initially considered adequate for representing HSTORESIS. This life-cycle model is *process-driven*, where the prototyping approach is utilized when the requirements are not well defined.
- o ARTIMIS (Automated Reporting, Tracking, and Evaluation Management Information System) seems to be capable of managing the schedules of a knowledge based system development project.
- o APCE (Automated Program Control Environment) provides a method to create and control a project instance. The author first enters a project plan and schedule. The interface (in version 3.1.1 of the SSE System) was sufficiently adequate to create a project instance with no real difficulty, and provided the creator with ability to tailor the project development phases. The APCE uses an *event-driven* control mechanism. Elements entered into the project instance are treated as a linear list of discrete events that require some formal notification in order to transition from one phase (e.g., requirement) to another (e.g., development). If it is necessary to make a modification to an element in a previous phase, this is treated as an anomaly that triggers some change control mechanism - at odds with a prototyping methodology where requirements analysis, design work, and coding are treated as an iterative process. Finally, APCE has no support for rules.

## **Recommendations**

- o **Iconix CASE tools:**
  - have some bugs which should be fixed
  - need better support of object-oriented designs
  - need graphical capability to distinguish between discrete and continuous data flows, and for representing a control transformation;
  - should provide a state transition diagramming facility, and an entity-relationship diagramming facility (analogous to Apollo's tool-set)
  - should be more closely integrated
  - should support automatic drawing of diagrams from a non-graphical specification
  - should have a flexible method of requirements-code traceability
- o **APCE should support additional software life-cycle models (e.g., prototyping) - study**
- o **Further study on tool versions for estimating KBS resource requirements**
- o **Further study on methods to represent rules in Iconix- or APCE-like systems**

## Recommendations

- o Iconix CASE tools:
  - have bugs which need to be fixed
  - need better support of object-oriented designs via real-time extensions in FreeFlow - add graphical capability fro distinguish among discrete and continuous data flows, and for representing a control transformation;
  - should provide a state transition diagramming facility, and an entity-relationship diagramming facility
  - should be more closely integrated so that changes at one level are correctly propagated to the next (in their current state that cannot support the interactive development of program requirements and design)
  - should support automatic drawing of diagrams from a non-graphical specification (e.g., generate an entity-relationship diagram from a list of entities and relationships)
  - should have a flexible method of linking requirements to affiliated source code for traceability
- o APCE should support additional software life-cycle models (e.g., prototyping) - further study is needed here (spiral model may not be appropriate)
- o Further study to develop tool versions to estimate resource requirements for AI implementations (FPA & COCOMO are inadequate as is)
- o Further study on methods to represent rules in Iconix- or APCE-like systems



**SSE/DKC**

## **Plans**

- o Disseminate report - "Hubble Space Telescope Design/Engineering Knowledge-base Software Support Evaluation"
- o Continue interface with SSE representatives
- o Continue interface with other DKC researchers/developers
- o Seek support for further investigations
- o Implement HSTORESIS as part of HSTDEK fault-diagnosis system for Hubble Space Telescope
  - in the Huntsville Operations Support Complex
  - in support of Orbital Verification
  - implementation will utilize SSFP-compatible hardware
  - partially supported by SSFPO (Level II)



SSE/DKC

## Participants

### Space Station Freedom Program Office:

- Dr. Mike Freeman/SSS,

### Marshall Space Flight Center:

- Tom Dollman/EB44
- Michelle Perrin/EL15

### Lockheed:

- Kevin Rawyers /LMSC
- Preston Cox/LMSC
- John Forbes/LAIC

### Stanford University - Knowledge Systems Laboratory:

- Dr. Robert Englemore, PI
- Post-Doctoral & Graduate Researchers

### Ames Research Center:

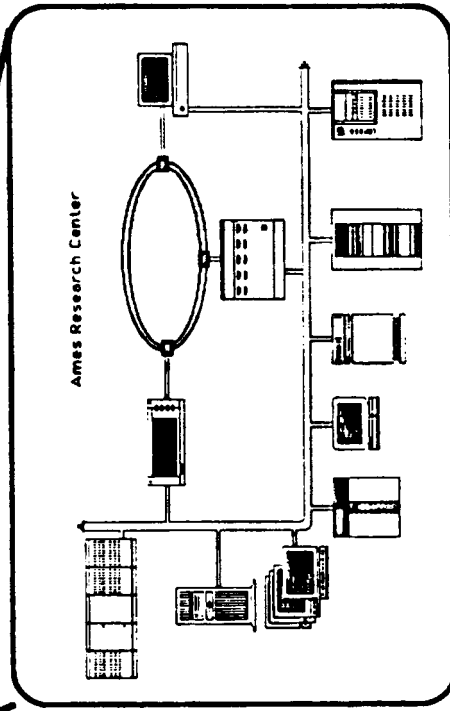
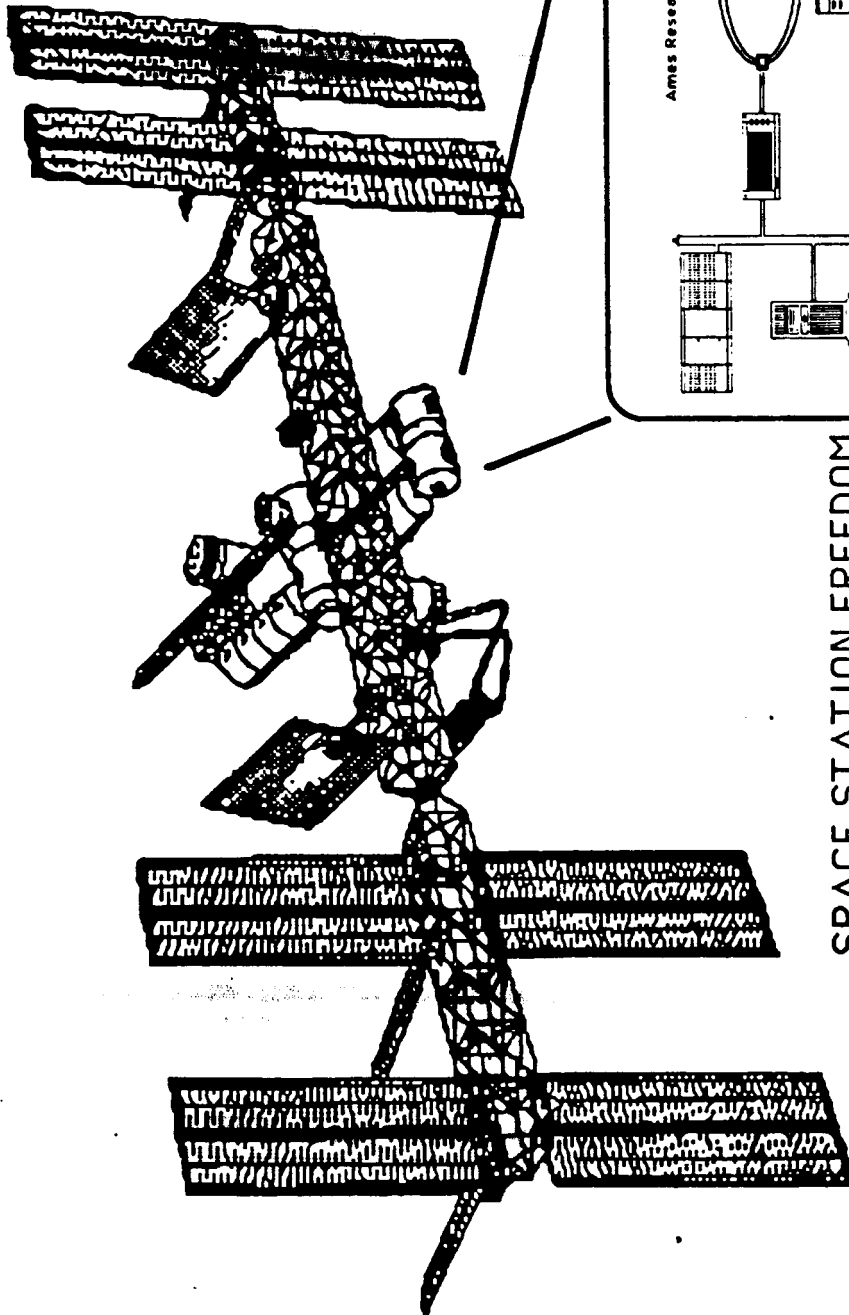
- Dr. Peter Friedland/RIA (COTR)

## **Summary**

- o Activities yielded initial assessment for using one SSE implementation on the development of a representative DKC model.
- o Study pointed up areas which require additional investigation
- o Initial year's effort beneficial to SSFP/SSE:
  - demonstrates methodologies on how DKC can be accomplished in SSE framework
  - identifies methods to test and validate telemetry linkages
  - identifies opportunities for effective upgrades
  - provides recommendations to feed into the PDR/CDR process

## Summary

- o The activities in FY89 yielded an initial assessment for using one SSE implementation in the development of a representative DKC model.
- o This study pointed up areas which require additional investigation, a jumping-off place for future work
- o The initial year's effort will be beneficial to SSFP/SSE. This project:
  - demonstrates methodologies on how DKC can be accomplished in SSE framework
  - identifies methods to test and validate telemetry linkages
  - identifies opportunities for software upgrades to improve effectiveness
  - provides recommendations to feed into the PDR/CDR process to accommodate other activities



SPACE STATION FREEDOM

DMS



# ADVANCED ARCHITECTURE TESTBED

AMES RESEARCH CENTER

## Advanced DMS Architectures Testbed - Mr. Terry Grant

### ABSTRACT

The objective of the Architecture and Tools Testbed is to provide a working, experimental focus to the evolving automation applications for the Space Station Freedom DMS. Emphasis is on defining and refining real- world applications, including the validation of user needs, understanding of system requirements and capabilities, and extension of capabilities. The approach is to provide an open, distributed system of high performance workstations representing both the Standard Data Processors and networks, and advanced RISC-based processors and multiprocessor systems. The system provides a base from which to develop and evaluate new performance and risk management concepts, and for sharing the results. Participants are given a common view of requirements and capability via: remote login to the testbed; standard, natural user interfaces to simulations and emulations; special attention to users manuals for all software tools; and E-mail communication. The testbed elements which instantiate the approach are briefly described including the workstations, the software simulation and monitoring tools, and performance and fault tolerance experiments.

# **ADVANCED DMS ARCHITECTURES TESTBED**

**BY**  
**Terry L. Grant**

**NASA AMES RESEARCH CENTER**

## **OVERVIEW:**

- \* CONCEPT: SHARED EXPERIMENTS for  
AUTOMATION INSERTION**
- \* HISTORY and PLANS PERSPECTIVE**
- \* TESTBED CONFIGURATION**
- \* SIMULATION MODELS**
- \* EMULATION SUPPORT EXECUTIVE**
- \* FUTURE EXTENSIONS & POTENTIAL**

## THE CONCEPT

The concept of an Advanced Architectures Testbed is a natural one for inserting automation into the Data Management System (DMS) onboard Space Station Freedom. A testbed is necessary to evaluate and integrate all the new user features of advanced automation and to create a high-performance system which can also be robust enough to be an acceptable candidate for the rigorous development, integration, and test procedures applied to real flight systems. The testbed implies "trial-and-error" experiments with applications on state-of-the-art computers and workstations with a multidimensional viewpoint: (1) operational evaluation of new multiprocessors and high-performance workstations; (2) productivity experiments with knowledge-based systems (KBS) in various domains and at various levels of functionality; (3) development experiments with various automated procedures and KBS; (4) definition and refinement of real-time requirements for Operations Management Application and driving user experiments; (5) investigation of failure detection, isolation, and recovery issues; and (6) the study of methodology for adapting to new requirements and new capabilities.

The challenge of the testbed for advanced architectures is to provide the infrastructure to ease experiment planning, execution, and analysis while at the same time facilitating communications of the problems and insights as they develop among the various designers, developers, and users who participate. The open sharing of methods and tools and development of a common understanding of objectives for DMS evolution will be as important to inserting new technology as the success of individual experiments.

AUTOMATION  
ADVANCED<sup>V</sup> ARCHITECTURES TESTBED

CONCEPT: An Experimental Focus  
to Automation Architectures

A collection of SOA computers & workstations

A multi-dimensional viewpoint:

H/W, S/W, Projects, Studies

An Infrastructure for sharing:

Tools, Interests, Problems, Insights

A dynamic vehicle to experiment with  
change itself!

## A HISTORY and PLANS PERSPECTIVE

About two years ago, when the Advanced Architectures Testbed was being planned, there was no defined testbed network configuration. The hardware consisted of the Sequent B8000 multiprocessor, a few varieties of symbolic workstations, and the division shared main frame, a VAX 8800. The cluster of SUN3 workstations were just arriving, and with them the first experiences under UNIX. Lisp and Ada programming languages were available for development of performance benchmarks and knowledge-based systems. The baseline (1/87) Data Management System (DMS) Architecture Control Document for the Space Station Freedom was available at the start and the Advanced Architectures development group at ARC was represented at the first DMS working group meetings. The first experimentation within ARC, and shared with the work package centers, has focused on the local area network performance and used simulation of the high-performance fiberoptic ring technology and the baseline Global network protocol, FDDI.

During this past fiscal year the testbed approach has been expanded on multiple fronts. (1) A redundant fiberoptic ring has been added and a network link provided to the Sequent B8000 multiprocessor operating system. (2) Three new RISC computers have been added, each with more than 10 MIP performance, the SUN4, the DEC 3100, and the MIPS M2000. (3) A 386i workstation represents the DMS standard processors for interim fault tolerance studies, and (4) a high-performance lisp workstation provides new KBS capability. (5) The LAN configuration, software, and projects are described in an experimental Hypercard configuration document. (6) Additional analysis of the FDDI protocol for typical core scenarios has been released, and (7) the Distributed Processor Network Simulator has been revised for further high-level scenario experiments. (8) A model of the Ancillary Data Service has been built for DMS performance studies. (9) Additional manuals have been distributed on our simulation tools, and user accounts provided to other groups working on the DMS. Several emulation experimental support packages have been added to the testbed: MEASURE - an integrated data-analysis and model-identification facility; FAUST - an environment for software mutation experimentation and analysis; ISIS - a tool-kit for distributed and fault-tolerant programming; and a Fault Tree Editor/Diagnoser. The lisp benchmarks for symbolic processing have been rewritten in Ada for performance evaluations in the standard Software Support Environment language. A detailed description of these tools is beyond the scope of paper, but an overview and contact for further information is included in the testbed documentation. The support package descriptions and the Hypertext testbed documentation are two examples of the methods being tried to encourage more participation, a necessary element for the success of technology evolution.

Future plans for the testbed include development in each of the cited areas and in particular: (1) the conversion of the redundant fiberoptic ring to the FDDI standard; (2) the addition of PS/2 workstations and DMS kit hardware as it becomes available; (3) the addition of i486 computers; (4) development of a prototype KBS to aid network testing; (5) development of a Fault Injection Automated Testing Environment; (6) addition of a common users interface; (7) integrated simulation and emulation experiments; and (8) the extended sharing of facilities, methods and results with a broad group of systems engineers and end users of Space Station Freedom.

# ADVANCED ARCHITECTURES TESTBED

## HISTORY/PLANS PERSPECTIVE:

### YESTERDAY

No Configuration  
Ethernet/DECnet  
& /TCP

### TODAY

< LAN Configuration >  
Redundant FO Ring  
/TCP  
NFS

### TOMORROW

FDDI Protocol  
? XTP - ISO

SUN 3's

TI CLM

Sequent B8000

LMI; VAX 8800

< Main frames & Workstations >

+ DEC 3100; SUN 4

IIM 45000;

MIPS M2000;

386i

Router GW

+ PS/2's; i486's;

DMS kit;

New RISC's

Simulation:

LANES; DPNS

Lisp Benchmarks

< Tools & Methods >

+ Emulation;

+ AXE

+ Measure

UDP Tests

Ada Benchmarks

+ ISIS; FAUST;

FaultTree Editor/

Diagnoser

+ NTPE; FIAT;

Common Users Interface

Integrated Simulation/

Emulation

Performance Reports

Reliability Reports

RII, GSFC

< Participants >

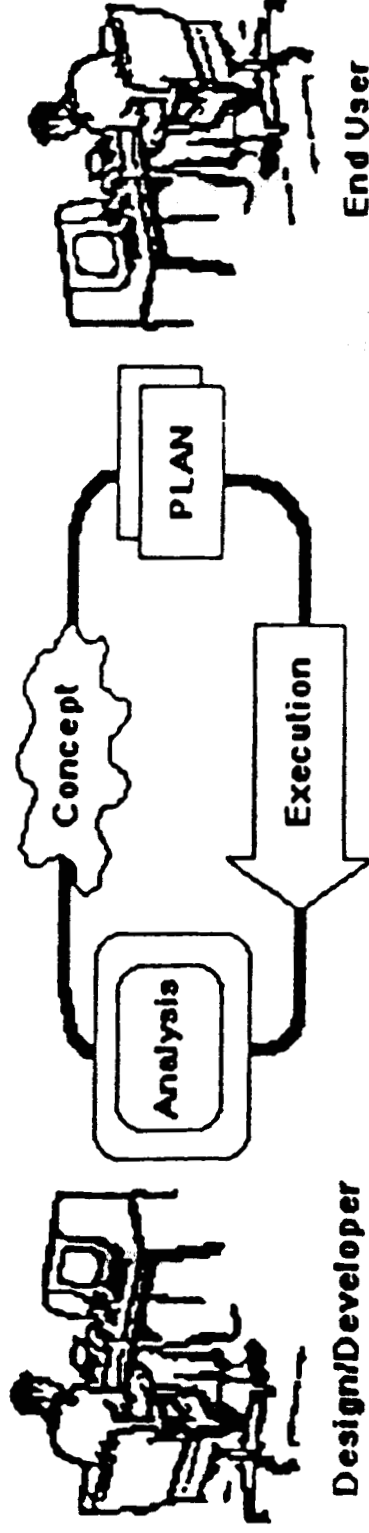
+ JPL, JSC / level-3 SSF

+ level-2 SSF; SS User

## HYPERCARD DOCUMENTATION of THE ADVANCED ARCHITECTURES TESTBED

The testbed description via Hypercard is an experiment in providing a detailed view of the computing power, software capabilities and projects in a more useable and readily adaptable form. Hypercard is a form of hypertext media which not only integrates text and graphic material, but also supports multiple ways of linking the individual objects or cards so that the user can direct the order of their presentation instead of the author fixing the order of presentation, as in conventional linear text. The cards in a Hypercard application are internally structured as a stack and thus the application is commonly referred to as a "stack", but the complete linkage is more complex and must be programmed to meet the needs of the application. The documentation experiment is open to comments and suggestions from all interested participants. The stack is planned to be reissued quarterly as changes and additions are made.

## Advanced Architecture Testbed Objective



### EXPERIMENTAL FOCUS for EVOLVING ADVANCED AUTOMATION TOOLS & ARCHITECTURES on NASA MISSIONS.

- Develop simulation & monitoring tools.
- Define/refine Real-World applications for fault tolerance and performance.
- Validate user needs, understand/predict system requirements & capabilities.

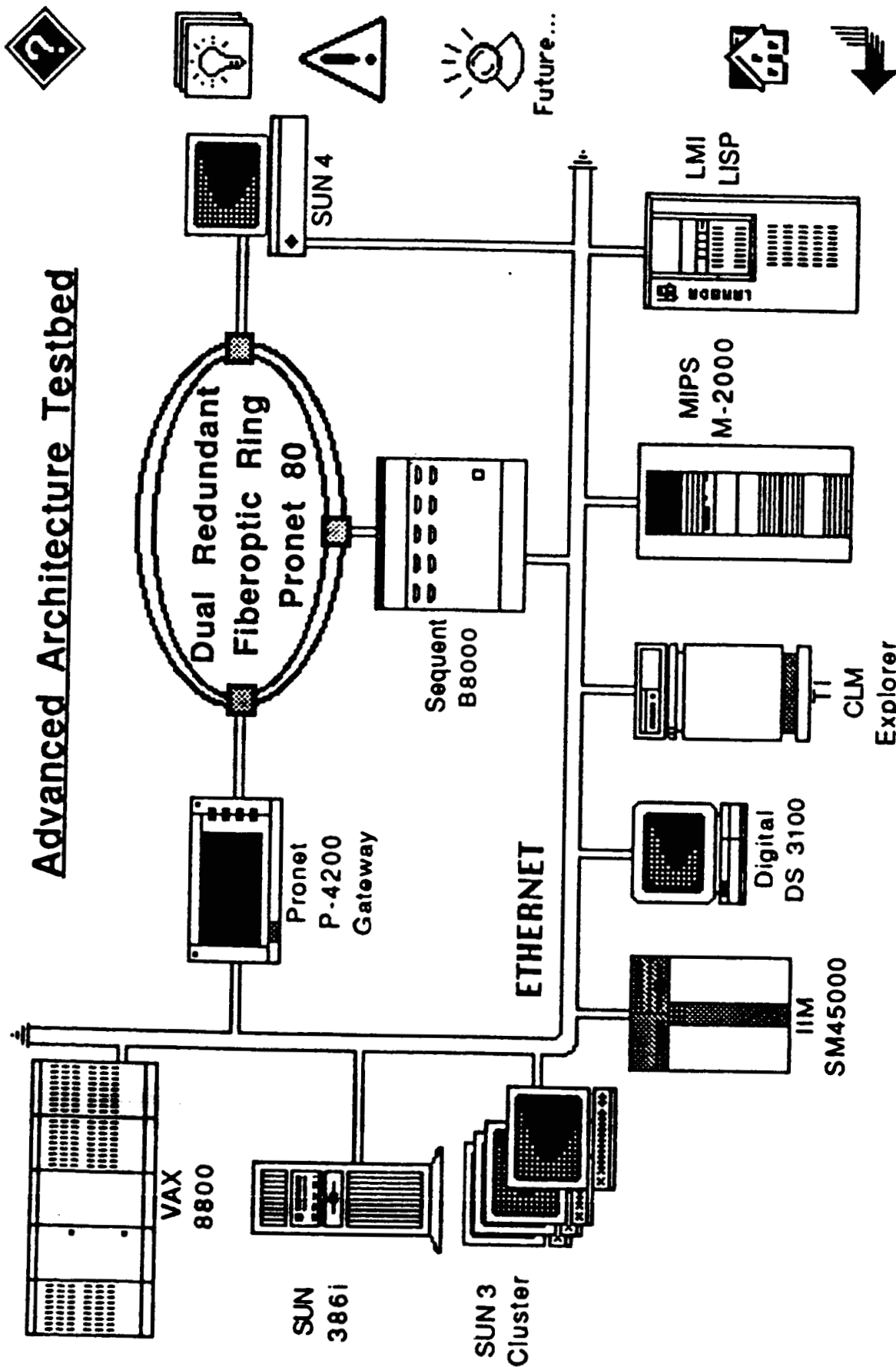
### • PROVIDE CLOSE TIES to the SPACE STATION FREEDOM DEVELOPMENT COMMUNITY by GIVING PARTICIPANTS a COMMON VIEW of REQUIREMENTS and FUTURE CAPABILITIES in INFORMATION SYSTEMS TECHNOLOGY.

- Provide remote login access as well as E-mail interaction with participants.
- Develop portable, well-documented software tools.
- Use a standard, portable user interface for simulations and tests.

## TESTBED CONFIGURATION

The Advanced Architectures Testbed Hypercard description is complete with an introductory help, for first time users; a viewpoint chooser with three perspectives: software, testbed configuration, and projects; and an objective overview. The figure shows the configuration viewpoint, illustrating both the computing power and the connectivity available. The user can "click" on the computer or network of interest to obtain more detailed information on the element of choice. Three RISC computers are available, the SUN4, Digital's DS 3100, and the MIPS M2000, each with more than 10 MIPS of raw computation power. Three high-performance symbolic workstations are also available for experimentation, the IIM SM45000, the TI CLM Explorer, and the LMI LISP. The SUN3's and SUN 386i are available for UNIX performance comparisons and support many special packages for fault tolerance studies also. The Sequent B8000 and SUN4 are connected to both Ethernet and the 80 Mbps fiberoptic ring for on-going network performance experiments. Note also the standard icon on the far right provide links other cards and provide control: at the bottom the backward arrow always links to the last card viewed before this one. The "home" icon links back to the home or top card. The "future" icon links to a card showing the configuration currently planned with more connectivity and added computing power. The exclamation point and light bulb icons go together for using and editing the card respectively; these won't be on the versions distributed to most users, but make it easy for a small group of developers to edit changes and corrections.

# Advanced Architecture Testbed



## TESTBED HARDWARE DESCRIPTION EXAMPLE

The SUN4 description is a typical example of the way a computer hardware element is presented in the Hypercard format. The network name for the workstation, "Goliath" is the first item in the upper lefthand corner, followed by the full ARPAnet address and then the two physical address numbers, one for the Ethernet link and the other for the Pronet ring link. The hardware information shows in a scrolling window to assure that the number of items isn't limited to the visible number of lines. Details on each item, if required, are available by "clicking" on the item named, and then a second scroll window provides them. In this case, the CPU is further explained as a SPARC processor which runs at 25 MHz, "and can perform up to 10 million instructions per second." This small window is removed by simply "clicking" on it. Note also two standard icons on the hardware card, the contact, and the vendor. These link to names, phone numbers, and E-mail addresses for more details. At the bottom-right the backward arrow always links to the last card viewed before this one. The hand-on-the-button matches the hand icon for the mouse (not shown in this card image) and links to the initial viewpoint chooser card at the top of the stack. The next icon is a small configuration layout and links to the testbed configuration card just discussed.

Goliath.....

goliath.arc.nasa.gov  
128.102.25.45 (Ethe  
128.102.96.15 (Pro

SUN 4

workstations and servers. SPARC stands for Scalable Processor ARChitecture, emphasizing its applicability to large as well as small machines. The SUN4/110 SPARC processor operates at a clock speed 25Mhz and can perform up to 10 million instructions per second.

### Hardware

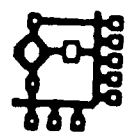
CPU: SUN 4100 CP  
FPP: SUN 4100 FPU  
Memory: 8 Mbytes S  
Network Interface: E  
Hard Disk: 330 Mbyte  
Tape Drive: 1/4" SCSI cartridge  
Monitor: 19" Color Monitor  
System Bus: VME Bus  
I/O Bus: Single-Slot VME Euro Standard  
Power: 115/230V, 50/60Hz, 12/6A

### Projects

Contact



Vendor



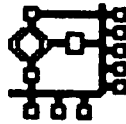
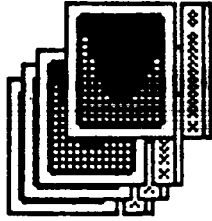
## TESTBED SOFTWARE DESCRIPTION EXAMPLE

This SUN3 description is a typical example of the way software for a computer element is presented in the Hypercard format. The software is categorized as Utilities, Languages, and Packages, with each item named and further information available by "clicking" on the name. Commercial software descriptions include vendor references, and special packages have contacts for particular detailed questions that may arise. This card on the software also allows access to the hardware or project information for the computer by "clicking" on the visible tabs to other cards in a natural analogy to real cards. This particular software list is fairly large, but not the complete testbed list; if you were interested in another language, such as Ada, you would just go to the "viewpoint chooser" and "click" on the total software list.

Ptolemy.....

SUN 3

ptolemy.arc.nasa.gov.....  
128.102.25.4 (ARC network).....  
128.102.112.1 (SUN Subnet).....

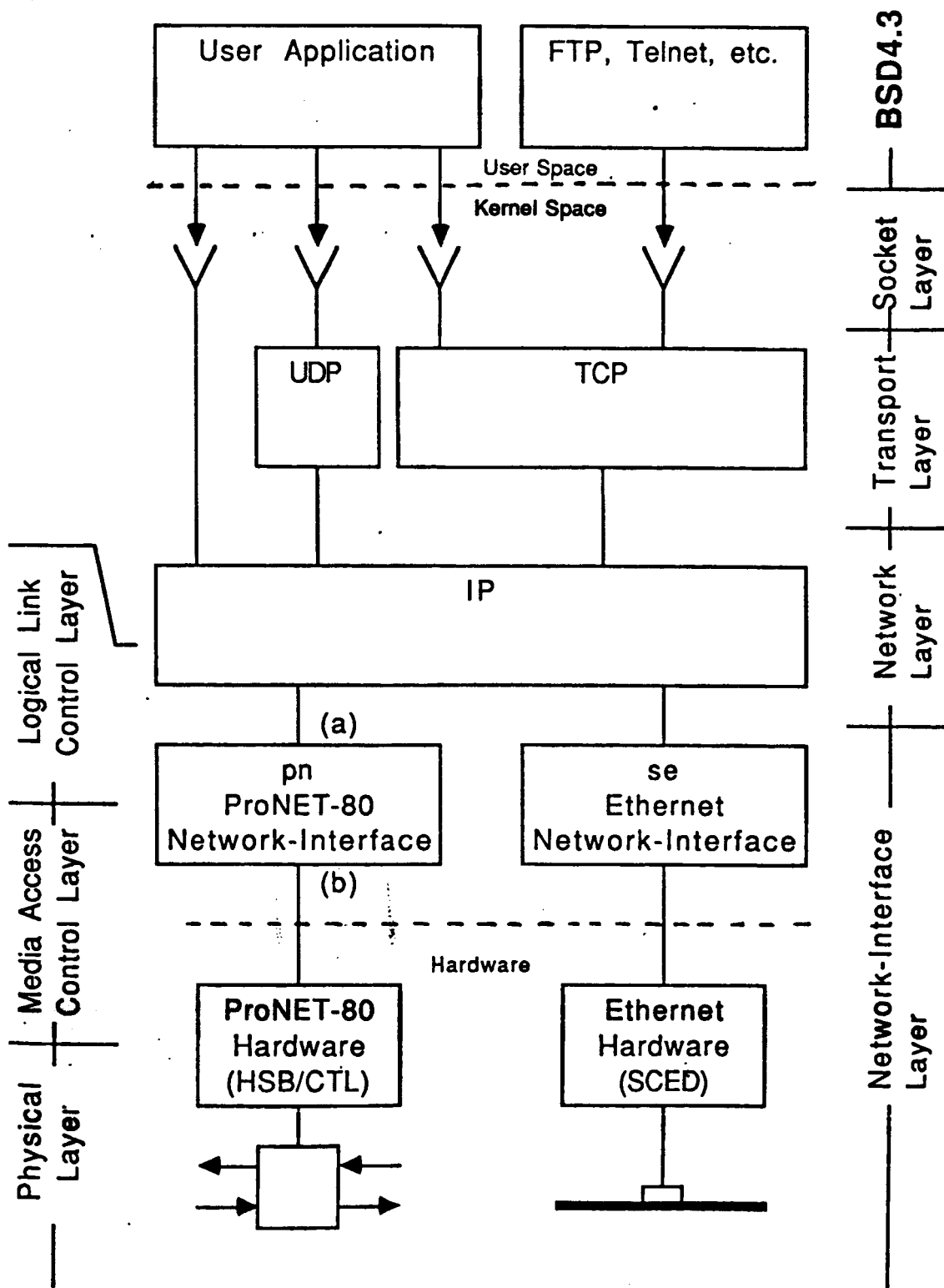


Hardware	Software	Projects
<b>Utilities</b> Sun OS 4.0 MS-DOS  TCP/IP  SunView X11R3 InterViews  Gnu Emacs	<b>Languages</b>  Sun C Gnu C Gnu C++ Franz Lisp Allegro Common Lisp	<b>Packages</b> FaultTree Editor FaultTree Diagnoser FAUST ISIS OBREL Measure

## SEQUENT B8000 NETWORK INTERFACES

A key requirement for our Advanced Architectures Testbed is the ability to experiment with real-time performance design options. No model of system performance is any better than the our knowledge of the performance of the individual elements and that knowledge is best verified by experiment. A major uncertainty in the performance of the DMS is capability of the network interface unit, or NIU, to link the Standard Data Processors via a high speed ring. Thus it has been important to have computers linked to a high speed ring for comparative tests relative to standard ethernet communications. The SUN4 workstation provided one such interface, but the second one (to the Sequent B8000) was developed inhouse this year to provide us with the in-depth knowledge of implementation issues. The interface hardware is handled on a secondary bus on the B8000 and provides an alternate port to the testbed network for experiments. The figure shows how the interface fits into the standard network reference model and where it sits in the UNIX kernel. The software connects the network-independent Internet Protocol (IP) module with the specific network implementation (Pronet-80). It provides hardware-specific initialization, packet transmission, packet reception, and statistics collection. The implementation uses 1600 lines of "C" source code and is fully documented. In order to support multiprocessor access on the B8000, it contains interlocks allowing concurrent execution by more than one CPU.

# SEQUENT B8000 NETWORK INTERFACES

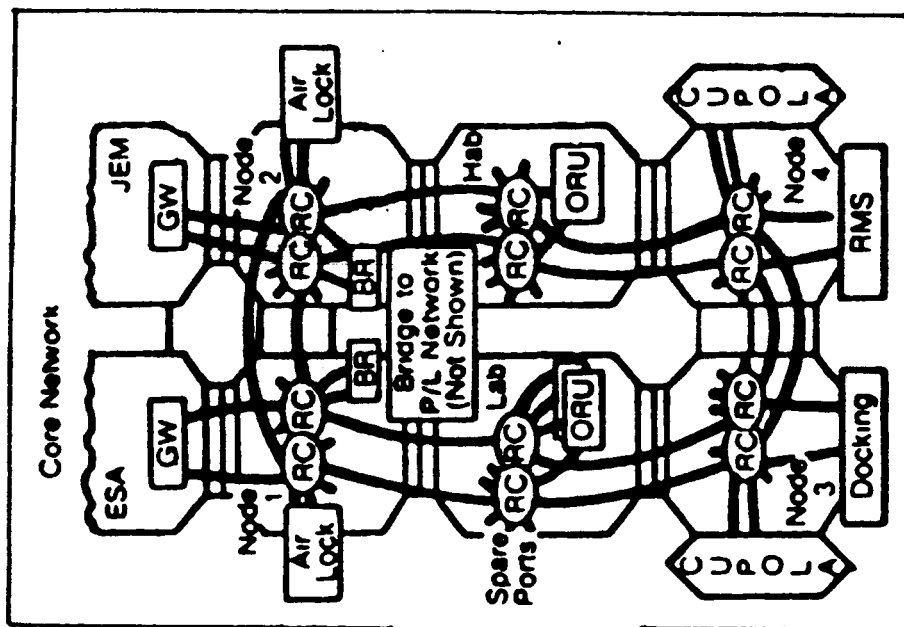
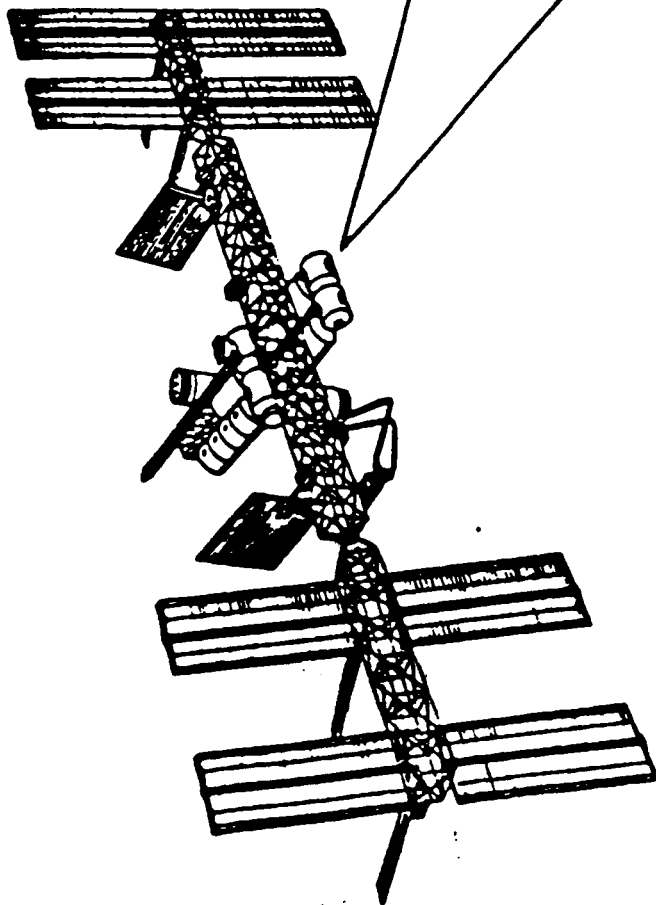


## AN EXAMPLE OF TESTBED SIMULATION FOR SSF

The usefulness of simulation for studying advanced local area network latency issues is summarized in the figure. The Local Area Network Extensible Simulator (LANES) on the testbed provides a method for simulating the performance of the high-speed Local Area Network link protocol Fiber Distributed Data Interface (FDDI), a ring network protocol which has been baselined for the Data Management System onboard Space Station Freedom. The appropriate technology for this network is now being developed, with breadboard and commercial beta test versions designed to transmit at high speeds using fiber optic cable. The FDDI model is based on the American National Standard FDDI Token Ring Media Access Control X3T9/84-X3T9.5/83-16 Rev.\ 10, February 28, 1986 (ANSI specification). The model is designed to determine performance characteristics of FDDI under a variety of loading and network buffering conditions. Source Code is written in Slam II; complete programmer and user manuals are available, and the code is listed with COSMIC.

# LATENCY STUDIES OF A HIGH-SPEED NETWORK PROTOCOL

## - APPLICATION TO SPACE STATION -



FDDI OPTICAL TOKEN RING PROPOSED FOR  
SS NODES & HABITAT - LAB MODULE

- NATIONAL/INTERNATIONAL DRAFT STD
- ISO-OSI 7-LAYER SUPPORT
- MANY MANUFACTURERS COMMITTED
- NO FDDI EXPERIENCE YET: COSTLY TESTBED
- DISCRETE EVENT SIMULATION FILLS THE NEED

## NETWORK TEST PROCEDURE DEVELOPMENT

The Network Test Procedure Executive (NTPE) is based on a KBS that integrates heterogeneous software for a design application. It is perhaps best understood in terms of the executive scenario (listed in the first figure) which it will automatically perform in response to a knowledge base on UNIX and the testbed configuration, together with a high-level test script. It is under development at the Boeing Advanced Technology Center, Seattle, by Dr. Kathryn Chalfun. The KBS was originally motivated by the need to formalize the problem-solving knowledge required to integrate and execute existing programs to produce a desired design analysis. It has been successfully applied to preliminary design problems for aerospace vehicles. Design analysis often requires that computer programs be integrated with a specific order-of-execution because of interdependencies. Due to the complexity of the interrelationships among the programs, numerous delays and errors often occur during their integration. By formalizing the problem-solving knowledge required to do the integration in a KBS, it is able to "understand" the objectives of the analyst and execute the correct programs in the correct order thus significantly shortening the design cycle.

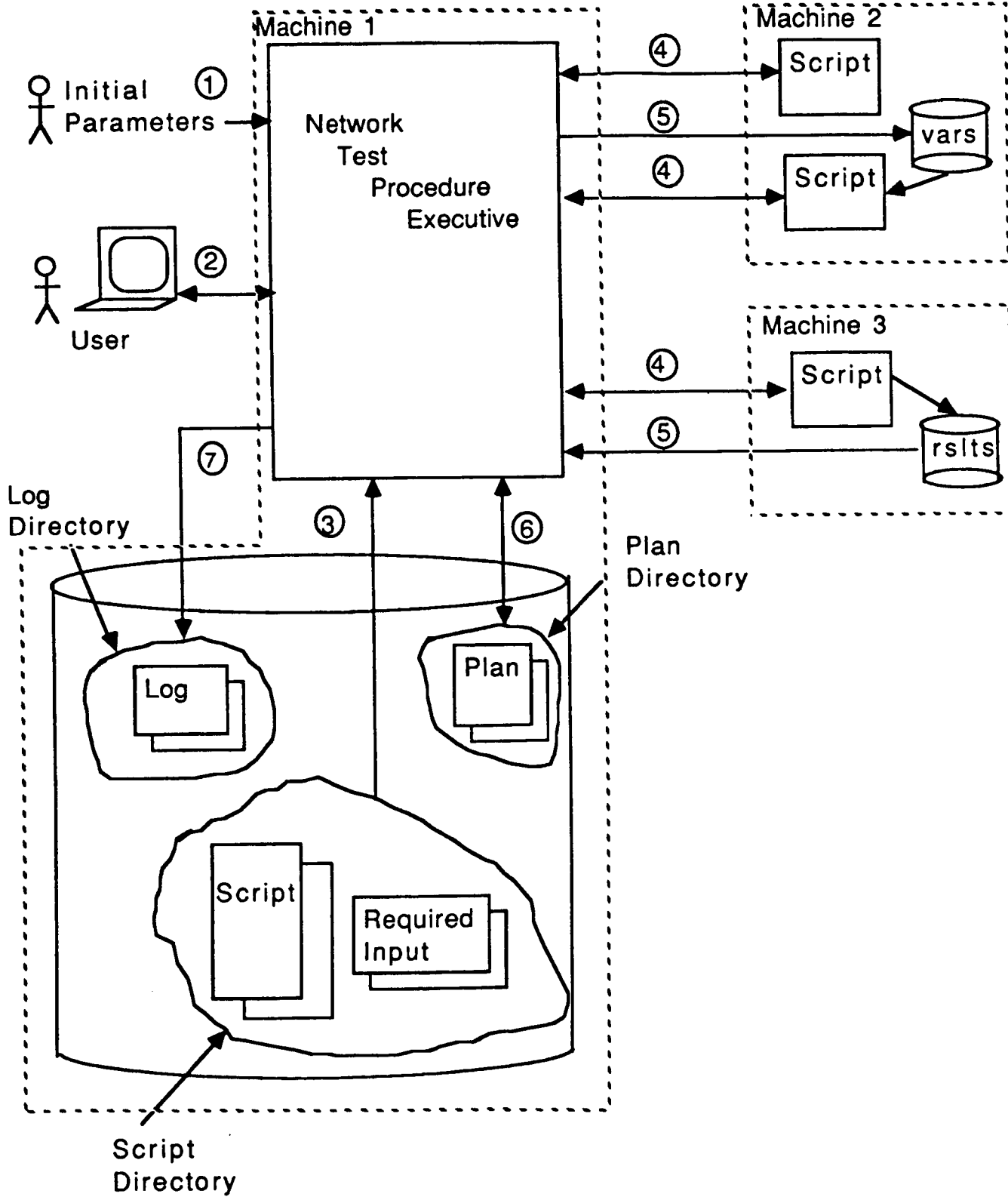
The NTPE prototype under development for the testbed will port to any UNIX workstation/computer and supervise UNIX utilities, workload processes, and monitoring/analysis tools for network development tests. We would like to broaden its potential and deepen our understanding of this KBS tool by trying to apply it to a management task of interest to the operations development at JSC, as well as to our testbed. The specification for the NTPE has been completed, and a prototype to explore usage issues is in process. The second figure gives an architectural view of its complexity by illustrating the many interfaces specified in its description. Initial C-shell scripts have also been written and tested according to the specification which demonstrate portable client and server functions.

## **NETWORK TEST PROCEDURE EXECUTIVE**

### **EXECUTIVE SCENARIO(S) - (RULE DRIVEN FROM OBJECTIVES)**

- Initiate background processing (locate benchmarks, validate configuration & processors, initiate interdependency script)
- Adjust activity levels (optional)
- Initiate monitors (locate, set parameters)
- Initiate foreground performance measure (Open measurement - record file, run-time, ...)
- Check run for completion (or for iteration-step completion)
- Initiate post-run analysis
- Signal "Test Complete" to User Interface

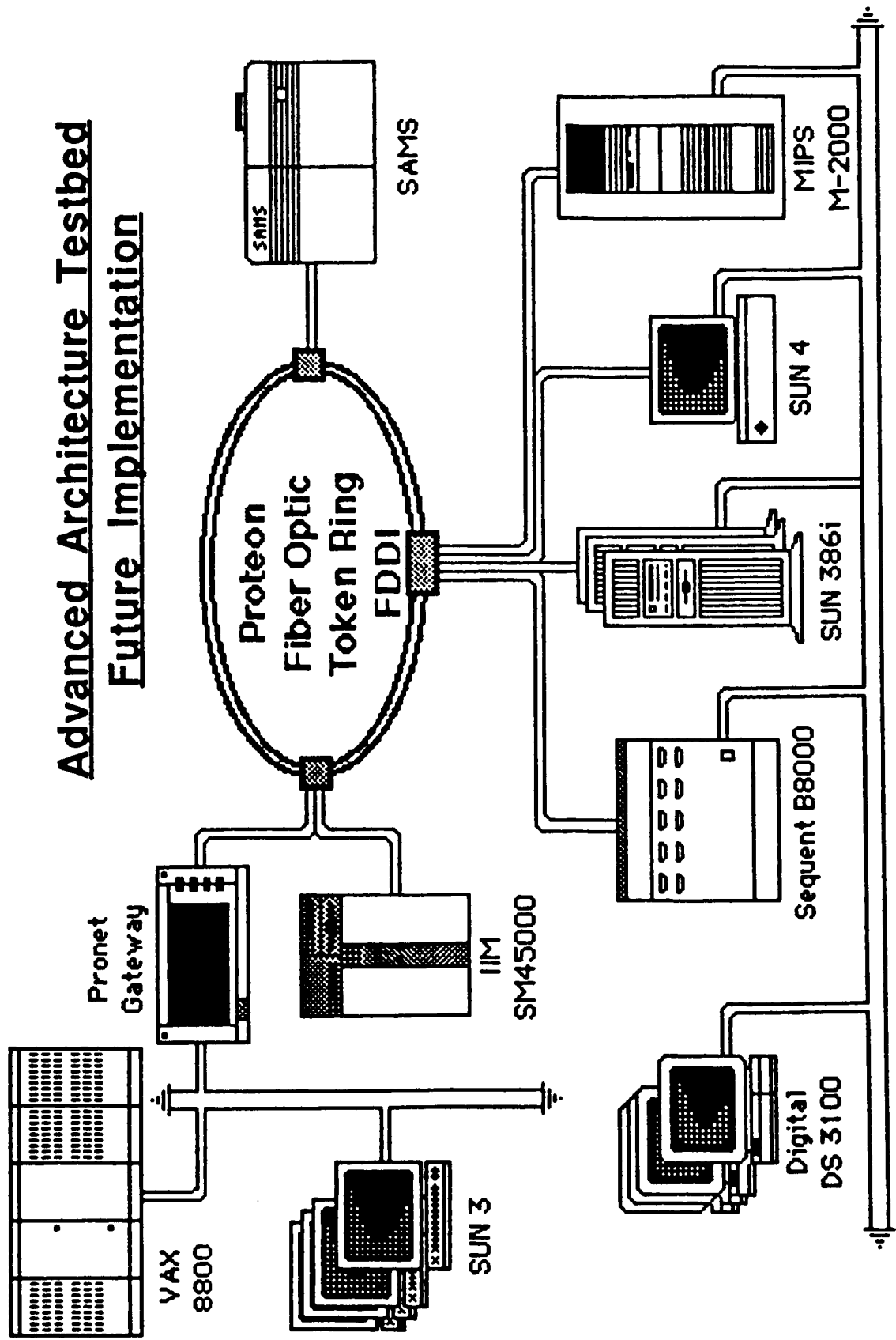
# NETWORK TEST PROCEDURE EXECUTIVE



## FUTURE TESTBED IMPLEMENTATION

In addition to the two individual development projects just described, many extensions to the current testbed configuration are probable in the near future. The figure illustrates some straight-forward extensions which will be done if they are encouraged by the DMS developers and system engineers as being helpful for adaptation to new technology and requirements: (1) the conversion of the Pronet 80 ring to the FDDI standard; (2) the addition of NIU's to other high-performance workstations; and (3) the addition of a spaceborne autonomous multiprocessor system and/or other OAST sponsored advanced processors. Of course many more projects can be added to the testbed, and the present ones have a variety of extension plans. The infrastructure developed at Ames will allow rapid evaluation of new KBS applications and processing architectures as well as rapid distribution of evaluations, resulting in better understanding of the readiness status of new technologies for insertion into the DMS.

# Advanced Architecture Testbed Future Implementation



# **SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS**

**UPN: 488-80-05**

**presented at  
Space Station Evolution Symposium  
League City, TX**

**February 8, 1990**

**by  
Dr. Ellen Ochoa**

**Intelligent Systems  
Technology Branch**



**Ames Research Center**

## SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS

### Objective:

Accelerate OAST-funded effort to study integrated numeric and symbolic processing requirements, and develop tools needed to integrate processors with DMS testbed.

### Motivation:

Baselined Space Station processors cannot support large real-time knowledge-based systems required for advanced automation.

Need to better understand user requirements for advanced automation, their computational characteristics, and the impact on the DMS.

Intelligent Systems  
Technology Branch



Ames Research Center

## SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS

### HISTORY OF TASK

- |           |   |
|-----------|---|
| FY88-89   | Co-funded by OAST and OSS to begin development of Spaceborne VHSIC Multiprocessor System (SVMS)   |
| Aug. 1989 | SVMS project cancelled by OAST because of lack of firm user requirements and shortage of funds  |
| FY90+     | Co-funded by OAST and Space Station Advanced Development Program to define user requirements, determine impact on computational resources, and develop tools needed to integrate high performance processors with DMS testbed |

Intelligent Systems  
Technology Branch



Ames Research Center

## SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS

### APPROACH

- Perform user requirements survey and hold Workshop on Advanced Automation Requirements (OAST-funded)
- Perform analyses of DMS elements and configuration to provide information on performance and options for upgrades
- Develop environments and tools for evaluation and integration of high performance processors in DMS testbed for advanced automation

Intelligent Systems  
Technology Branch



Ames Research Center

## **SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS**

### **AUTOMATION AND ROBOTICS REQUIREMENTS STUDY**

Contact: Gloria Davis (415) 604-4858

- Survey of mission planners/managers and technology developers  
automation goals of mission, degree to which objectives keyed to a specified computational platform, real-time requirements, details of current/planned processor system, spaceborne constraints, computational power required
- Advanced Automation Requirements Workshop at Ames  
(tentatively scheduled for May) with special sessions on computational needs  
Participants to include JSC and major aerospace companies
- Report with summary of data, computational needs charted against mission schedules, current or planned processing systems, areas where computational needs are not being addressed

**Intelligent Systems  
Technology Branch**



Ames Research Center

## SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS

### INTEL 386 AND i486 COMPARISON STUDY

Contact: Y.K. Liu (415) 604-4830

- Preliminary study in FY89 summarized key features of 486 and made recommendations on issues for further study

#### i486 advantages

2-4 times faster  
longer software life cycle  
multiprocessor support  
fully upward compatible from 386

#### Issues for further study

risk, quality, and reliability concerns  
power consumption  
radiation hardening

- FY90 plan is to evaluate performance using several benchmark suites on an IBM PS/2 486 with Lynx O/S, and meet with Intel and IBM to discuss radiation hardening and reliability issues

Intelligent Systems  
Technology Branch



Ames Research Center

## **SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS**

### **TIMELY NETWORK RESPONSE FOR FAULT MANAGEMENT**

Contact: Dr. Marjory Johnson (415) 604-6922

- Three published studies in FY89 on how various network protocols (FDDI media access control protocol and CCSDS protocols for advanced orbiting systems) can support Space Station applications requiring low latency
- FY90 work will develop strategies for assuring timely network response for fault management of the DMS

Model OMA contingency scenarios on Ames Advanced Architecture testbed, determine design and operational constraints of DMS/OMA baseline, measure response time and throughput on testbed using different strategies, identify and work to eliminate distributed system bottlenecks

**Intelligent Systems  
Technology Branch**

**NASA**

Ames Research Center

## SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS

### TOOLS FOR EVALUATION AND INTEGRATION OF HIGH PERFORMANCE PROCESSORS

- Developed an Ada Mathpack to allow Ada benchmarks to be ported among many systems

Includes transcendental functions and complex arithmetic. Assumes IEEE 754-1985

single binary floating point arithmetic and 32-bit integers. Contact: Dave Galant (415) 604-4851

- Developed a benchmark suite that includes scalar and parallel numeric and symbolic codes to measure performance of uni- and multiprocessors
- Demonstrated linear speedup in up to 10 processors for Parallel Ada benchmarks running on a multiprocessor Contact: Andy Goforth (415) 604-4809
- Developed an environment (AXE) which graphically illustrates parallel codes running on user-specified multiprocessing architectures

Contact: Dr. Jerry Yan (415) 604-4381

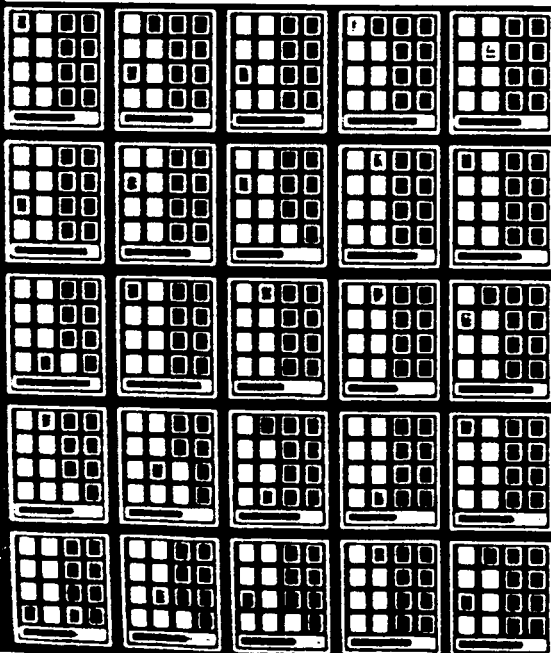
**Intelligent Systems  
Technology Branch**



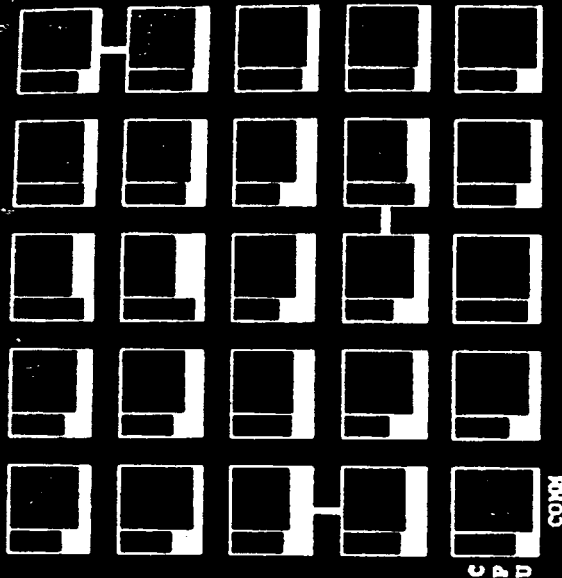
Ames Research Center

# AXE Experimentation Environment for Concurrent Systems

Process Status Panel



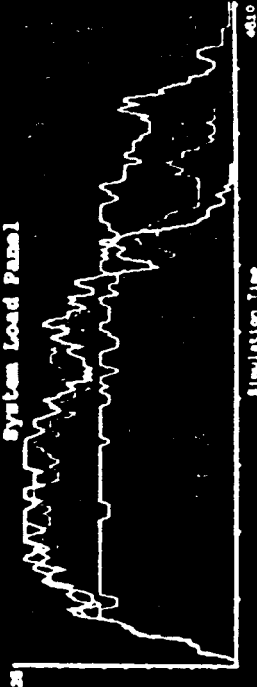
Multiprocessor Activity Panel



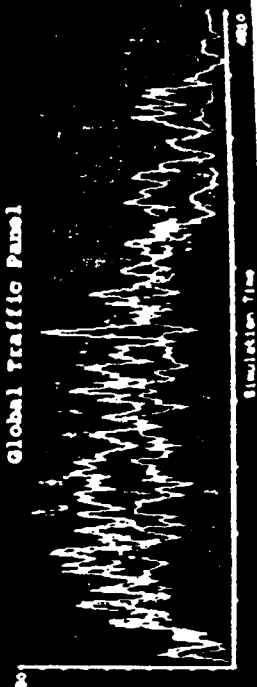
MOVE... mg:(21) <60>:  
 MOVE... mg:(22) <80>:  
 MOVE... mg:(23) <7>:  
 MOVE... mg:(24) <92>: [10] -> [18]  
 MOVE... mg:(25) <176>: [14] -> [8]  
 MOVE... mg:(26) <172>: [10] -> [24]  
 MOVE... mg:(1) <31>: [24] -> [10]  
 MOVE... mg:(2) <18>: [23] -> [4]  
 MOVE... mg:(3) <1>: [4] -> [14]

TIME:  
 940

System Load Panel



Global Traffic Panel



# PARALLEL ADA ON MULTIPROCESSORS

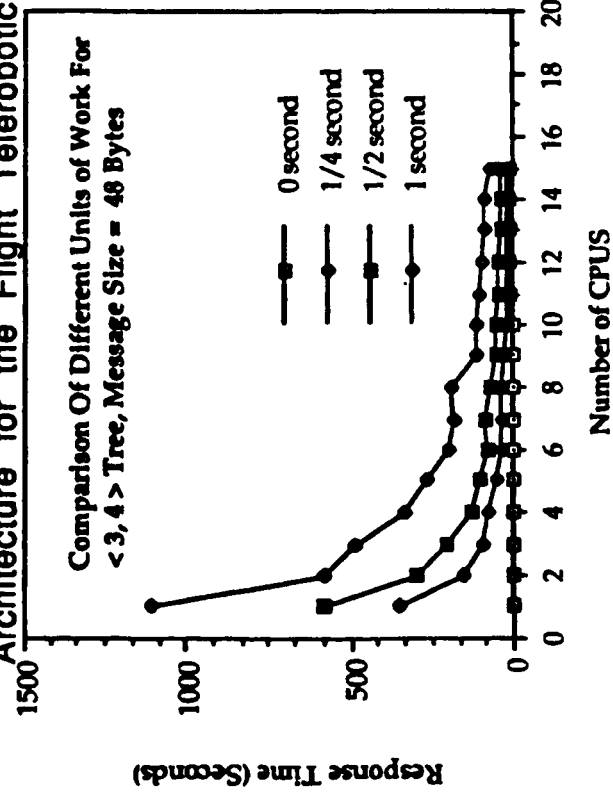
## RESULTS

- Performance efficiency is close to linear
- Scalability for another 10X improvement is currently possible

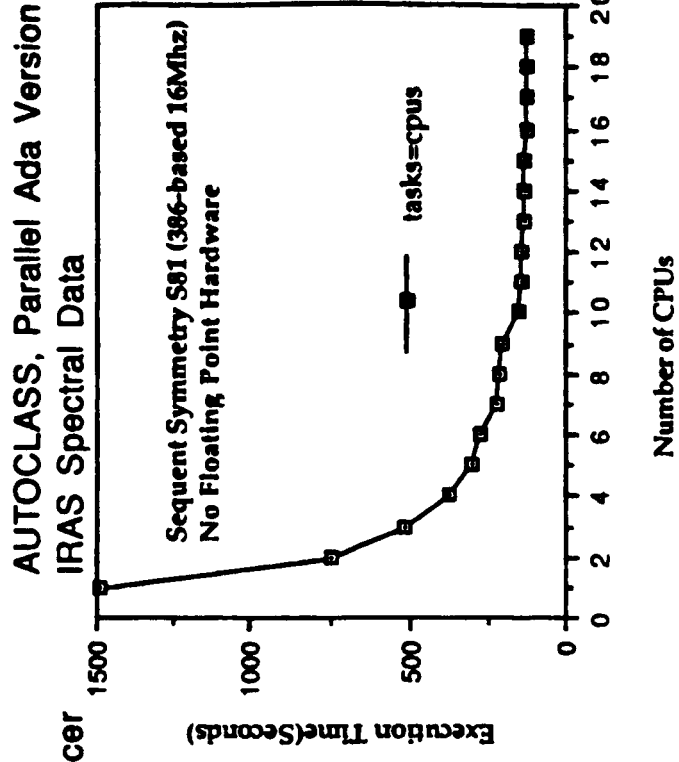
## AUTOMATION APPLICATIONS

### Telerobotic Control

Emulation of Hierarchical Control in NASREM Architecture for the Flight Telerobotic Servicer



### AI Classification



## **SPACEBORNE AUTONOMOUS MULTIPROCESSOR SYSTEMS**

### **FUTURE TECHNOLOGY CHALLENGES**

- Software infrastructure for real-time distributed multiprocessors, particularly with integrated numeric and symbolic capability
- Real-time fault management of multiprocessing systems, with built-in testability
- Languages, compilers, and translators for optimum development and operation

**Intelligent Systems  
Technology Branch**



Ames Research Center

**EOS PROCESSOR EXPERIMENT  
(ISES)**

**488-80-07-01**

**LANGLEY RESEARCH CENTER**

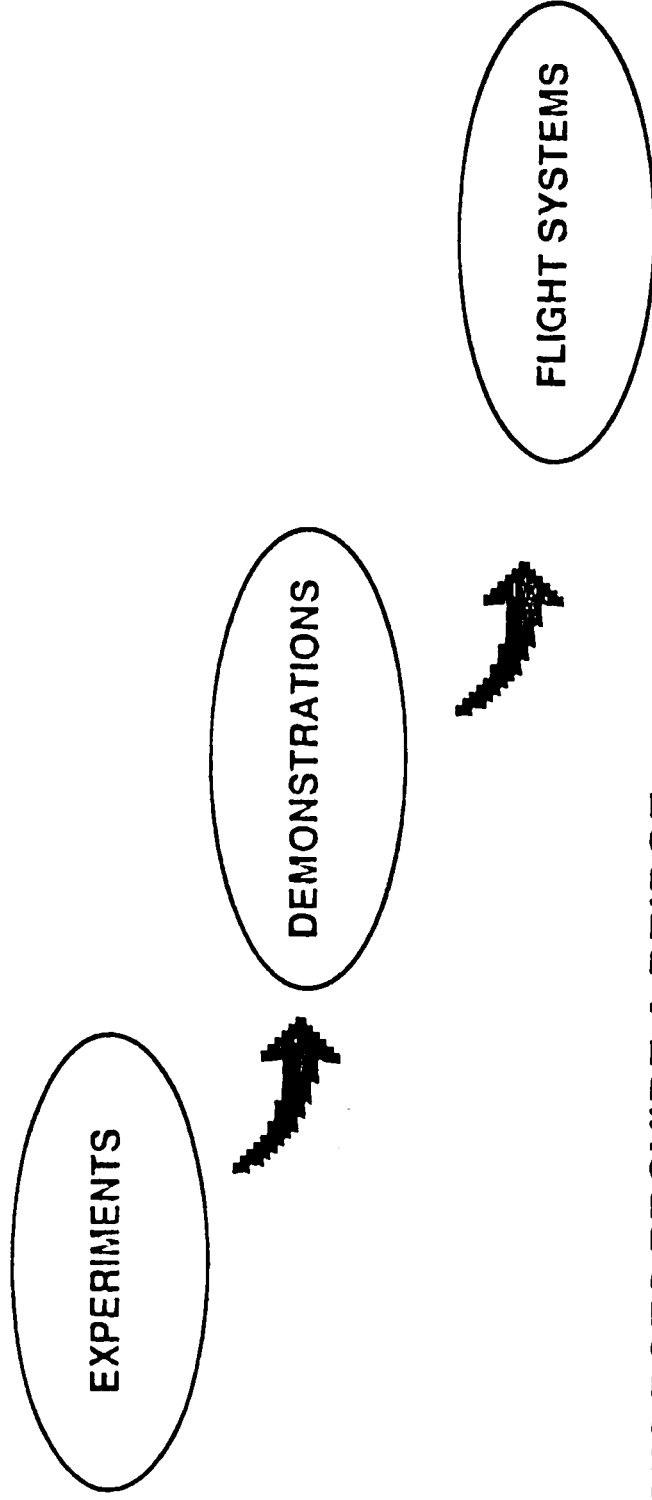
## **THE ISES CONCEPT**

- PERFORM ADVANCED ON-BOARD INFORMATION EXTRACTION EXPERIMENTS AND DEMONSTRATIONS USING REAL-TIME MULTISENSOR DATA IN SUPPORT OF Eos SCIENCE
- OPERATE IN A NON-INTRUSIVE EAVESDROPPING MODE ON OTHER INSTRUMENT DATA AND PRODUCE REAL-TIME SELECTABLE SCIENCE INFORMATION
- FUNCTION AS A LOW-DATA-RATE, LOW-DATA-VOLUME Eos EXPERIMENT CAPABLE OF BI-DIRECTIONAL DATA / COMMAND FLOW
- EVALUATE ADVANCED DATA SYSTEM AND COMMUNICATIONS TECHNIQUES AND TECHNOLOGY IN INFORMATION SCIENCE

The ISES Concept represents a new approach to rapid technology insertion in the information systems area. The key to the ISES concept is the commitment by the flight project of some "real estate" on the spacecraft--power, weight, access to the data and communication system, etc.--, dedicated to an information system "black box". Attached to the spacecraft data bus or LAN, the "black box" appears to be just another remote sensor, but in reality is a programmable computational resource which eavesdrops on the data network, copying data and producing selectable science information back on the data network. Being non-intrusive, the black box does not change or modify the main data stream.

## ISES VIS-A-VIS DMS

ISES DOES NOT PROPOSE TO "FIX" ANYTHING



ISES DOES PROVIDE A BRIDGE  
FOR RAPID TECHNOLOGY INSERTION

Capability growth can come from a variety of sources, not the least of which is the application of new technologies and techniques. Unfortunately, space flight systems rarely have provision for excess capacity (data weight, power, etc.) to permit accommodation of additional functions. Without the opportunity to evaluate potential technologies and techniques in space, the necessary "confidence-building" to convince program managers and project engineers to use new technologies or techniques is missing. New technology is rarely used because it is not mature; and by the time it gets mature, it is no longer new.

The ISES has as its objective on-board information extraction for use either directly to the ground or on-board. The ISES approach can effectively (1) address the insertion of new technologies and techniques without adversely impacting overall system reliability, (2) execute experiments and demonstrations leading to future operational systems, and (3) identify new high pay-off technology areas for development. The ISES approach does not propose to "fix" anything in the existing flight system. It does propose to impact future missions and systems.

## **SOME EXAMPLE ISES APPLICATIONS**

- 0 BETTER UTILIZATION OF ON-BOARD RESOURCES**
- 0 OPTIMIZED INSTRUMENT PERFORMANCE (E.G., LOOK AHEAD)**
- 0 MULTI-INSTRUMENT SENSOR FUSION/INFORMATION  
EXTRACTION**
- 0 REAL-TIME RESPONSE TO EVENTS, TARGETS OF  
OPPORTUNITY, ETC.**
- 0 QUICK-LOOK SCIENCE FOR BROWSING**
- 0 DIRECT SUPPORT FOR FIELD OPERATIONS**

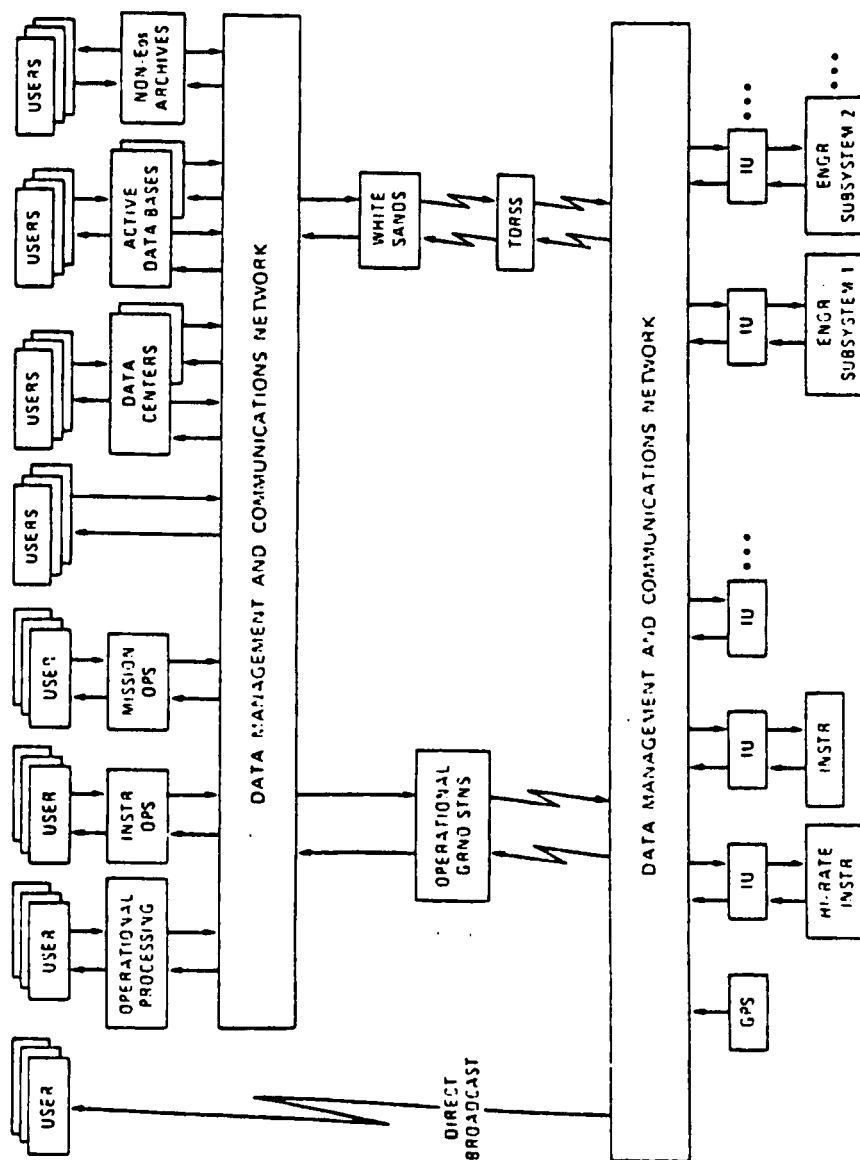
An on-board computational resource using the ISES concept can perform information extraction experiments and demonstrations in several important areas of systems operations.

## **ISES APPLICATIONS**

- o FOR THE POLAR PLATFORM ISES ENABLES ON-BOARD REAL-TIME, INFORMATION EXTRACTION FOR EARTH SCIENCES.**
- o FOR THE CORE STATION ISES SUPPORTS NOT ONLY EARTH AND PLANETARY SCIENCES, BUT ADVANCED CONCEPTS FOR MORE GENERALIZED SCIENTISTS AND USERS.**

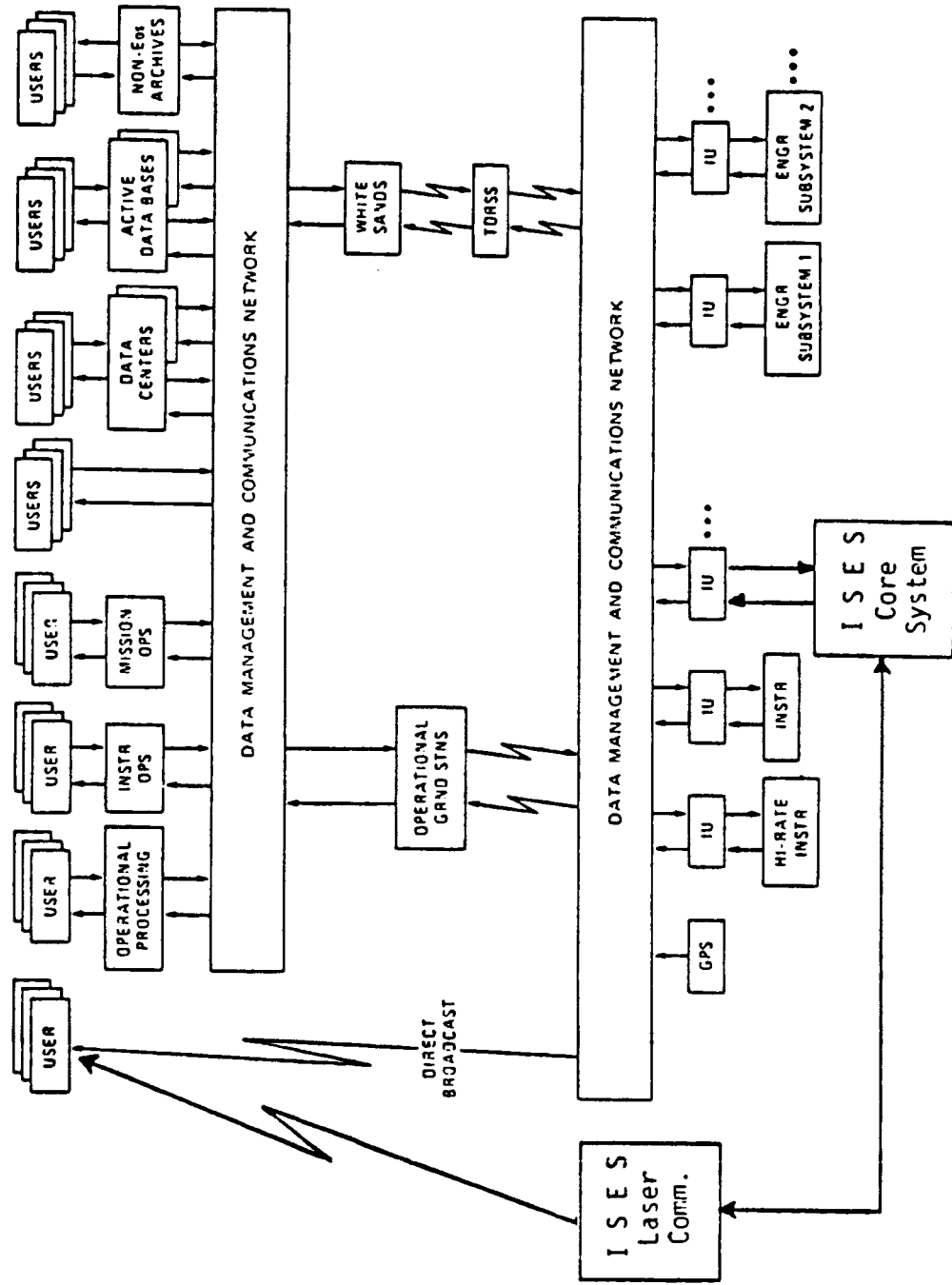
Because of the commonality of hardware and DMS concepts, the hardware developed for ISES can readily be adapted to either the Polar Platform or the Core Station. Whereas the Polar Platform is almost exclusively planned to support Earth and space sciences, ISES on the Polar Platform would support these objectives. The Core Station on the other hand represents a much more broad-scoped science and engineering opportunity to apply advanced information science and systems technology.

# A CURRENT Eos DATA SYSTEMS CONCEPT PRACTICAL DIMENSION



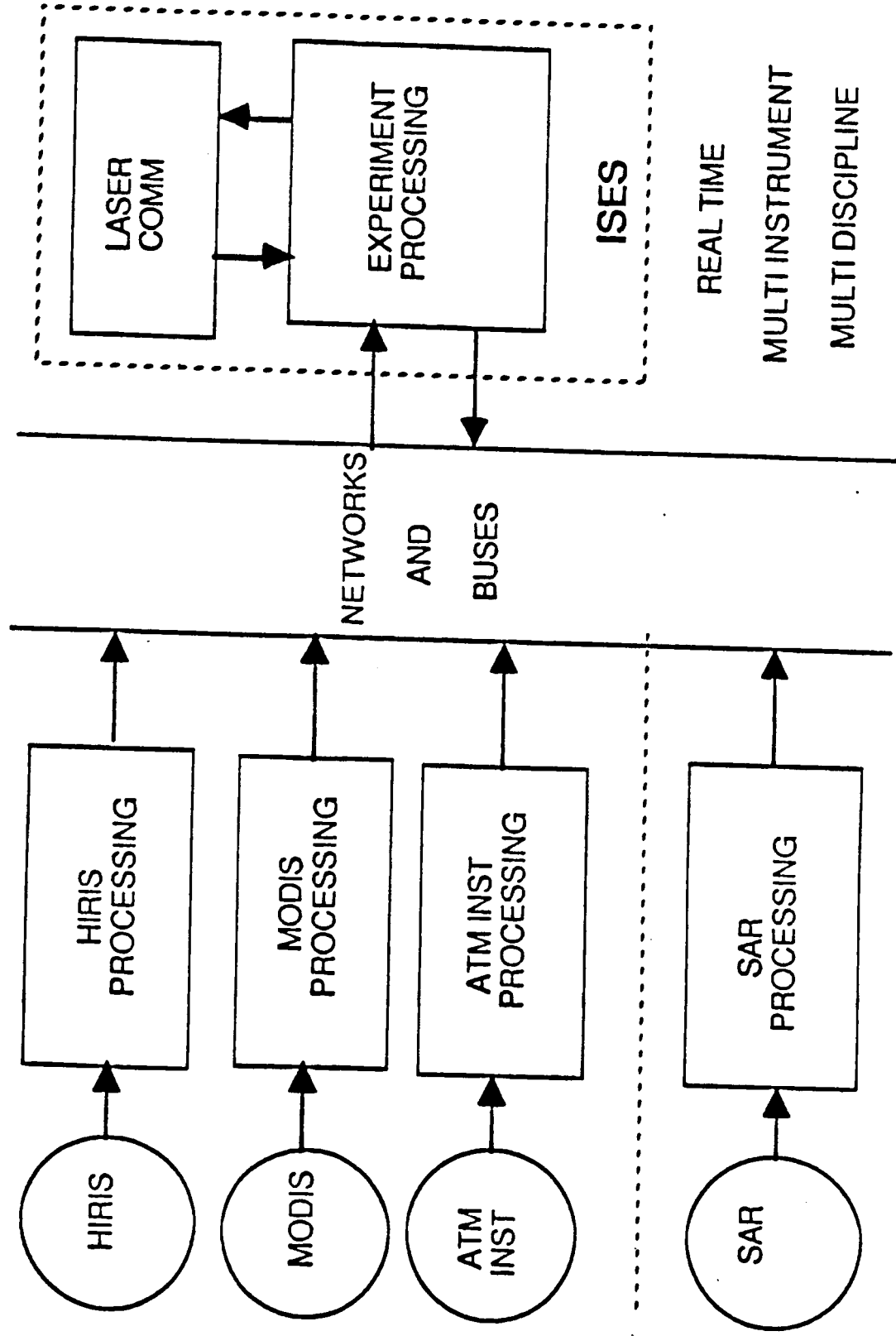
# INFORMATION SCIENCES EXPERIMENT SYSTEM

## PRACTICAL DIMENSION



This chart illustrates, in schematic form, how the ISES would fit into the EOS data environment.

# EOS INFORMATION PROCESSING ARCHITECTURE



The ISES is compatible with initiatives to perform "internal-to-instrument" processing for data editing, on-board processing, etc. ISES can utilize data or information at whatever level the Principal Investigators are willing to process. ISES, however, accepts and copies whatever already exists and does not modify instrument data in any "in-line fashion. This illustrates the "non-intrusive but involved" approach for ISES.

# THE SCIENCE TEAM AND THE TECHNOLOGY TEAM

## SCIENTIFIC AND TECHNICAL

- o THE SCIENCE TEAM DEFINES THE ON-BOARD EXPERIMENTS
  - AN EARTH SCIENCES GROUP MADE UP OF RESEARCHERS AND APPLICATIONS USERS WHO IDENTIFY THE EXPERIMENTS AND DEMONSTRATIONS REQUIRING ON-ORBIT REAL-TIME INFORMATION EXTRACTION
  - AN INFORMATION SCIENCES GROUP MADE UP OF INFORMATION SCIENTISTS WHO IDENTIFY EXPERIMENTS AND DEMONSTRATIONS FOR AN IN-SPACE REMOTE COMPUTATIONAL FACILITY
- o THE TECHNOLOGY TEAM DEFINES, DEVELOPS AND DELIVERS THE "BLACK BOX"
  - ADAPTS THE SCIENCE EXPERIMENTS TO FLIGHT SYSTEMS
  - DEFINES THE ISES FACILITY HARDWARE, SOFTWARE AND ARCHITECTURES

The pre-Phase A work supported under this task focussed on two major issues: (1) determining the scope of the science applications for ISES, and (2) assessing the major interface issues that the ISES "black box" would face on the Polar Platform and Core Station. To that end an in-house Science Team development was initiated with Earth Science and Information Science sub-groups. Studies were also started to delineate both the Polar Platform and Core Station on-board DMS and to identify the access points for data eavesdropping.

# SOME REAL-TIME DATA NEEDS FROM ISES EARTH SCIENCES WORKSHOP

WILLIAMSBURG, VA

MAY 1-4, 1989

## GEOLOGY

- o EARLY WARNING DETECTION
- o RAPID IMPACT / DAMAGE ACCESS

## ATMOSPHERIC SCIENCES

- o DETECTION OF EVENT TRIGGERS; E.G., STRATOSPHERIC WARMING AND CO2 OUTFLOW
- o ATMOSPHERIC ALERTS OTHER THAN VOLCANIC GASES; E.G., OZONE, CO, INDUSTRIAL POLLUTION, ETC.

## OCEANS

- o CHLOROPHYLL DATA
- o OCEAN BOUNDARIES

## COSTAL ZONE

- o ALGAE BLOOMS
- o ESTUARINE TRANSPORT; E.G., OCEAN DUMPING, OIL SPILLS, SEDIMENT RUNOFF, ETC.

## VEGETATION

- o CROP MANAGEMENT, E.G., IRRIGATION, SENESENCE, POLLUTION, ETC.
- o TRANSIENT EVENTS; E.G., CROP DRYING HOURS, FLOODING, FOREST FIRES

## SNOW, ICE AND SEASTATE

- o SEA / ICE BOUNDARY
- o ICE / LEADS RATIO FOR METEOROLOGICAL MODELS OF POLAR AREAS
- o SNOW / RAIN RATIO IN STORMS

## METEOROLOGY

- o LARGE STORM WIND FIELDS; HURRICANE EYE NOT ALWAYS CENTER OF STORM
- o NOCTILUCENT, HIGH CIRRUS AND CIRRUS CLOUDS; IMPORTANT FOR SHUTTLE RE-ENTRY
- o CLOUD INVENTORY FOR METEOROLOGICAL MODELING, TRANSPORTATION, ETC.

## INSTRUMENT SCIENCE

- o DECISION TO ACQUIRE DATA; E.G., USE OF LIGHTNING STRIKES FOR NITROGEN OXIDE STUDY
- o DECISION NOT TO ACQUIRE DATA; E.G., TURN OFF LASER OVER CLOUDS
- o ACQUIRE DATA FOR IMPROVED QUALITY; E.G., UP POWER FOR BETTER SIG / NOISE
- o COMBINING DATA FOR ENHANCED QUALITY; E.G., USE OZONE AND AEROSOL DATA TO IMPROVE MODIS DATA INTERPRETATION

The process of driving out Science requirements began, with an Earth Science Workshop in Williamsburg, Virginia, on May 1-4, 1989. Several major science discipline areas were chosen to "span-the-space" and a list of applications for "near-real-time" information extraction and "direct-to-user" services were identified.

The output of this workshop, currently being published as a Conference Proceedings, then becomes the initialization to the technology team as well as the lead in detailed definition and assessment of the applications.

# **SOME GENERAL OBSERVATIONS FROM THE ISES**

## **EARTH SCIENCES WORKSHOP**

- REMOTE SENSING OPERATIONS EMPLOYING FIELD SAMPLING TEAMS CAN GENERALLY BENEFIT FROM REAL-TIME SATELLITE DATA, INCLUDING QUICK-LOOK DATA.
- HIGH RESOLUTION IMAGING CHANNELS ARE VERY IMPORTANT FOR REAL-TIME OPERATIONS BUT DO NOT NEED TO BE TRANSMITTED ROUTINELY.
- DURING SUPPORT FOR FIELD EXPERIMENTS THE ON-BOARD COMPUTATIONS MAY BE INTENSIVE BUT ONLY NEED TO BE DONE OVER A RESTRICTED DATA SET AND AREA.
- ISES CAN PROVIDE BENEFITS FROM DIRECT ACCESS TO THE INSTRUMENT OUTPUTS AND NOT JUST THE LOCAL AREA NETWORK.
- THERE SHOULD BE A THOROUGH ANALYSIS MADE OF EACH INSTRUMENT TO BETTER UNDERSTAND ITS CAPABILITIES AND HOW IT CAN INTERACT WITH AND SUPPORT OTHER INSTRUMENTS.
- THE REAL-TIME DATA NEEDS ESTABLISHED BY THE VARIOUS PANEL MEMBERS SEEM TO HAVE BEEN LIMITED SOMEWHAT BY THE INDIVIDUAL SCIENTISTS' EXPERIENCES AND INTERESTS. THIS SUGGEST THAT ADDITIONAL PANEL EFFORTS WILL YIELD EVEN MORE PROVOCATIVE REAL-TIME DATA NEEDS.

Listed here are some general observations from the workshop concerning the role of on-board information extraction.

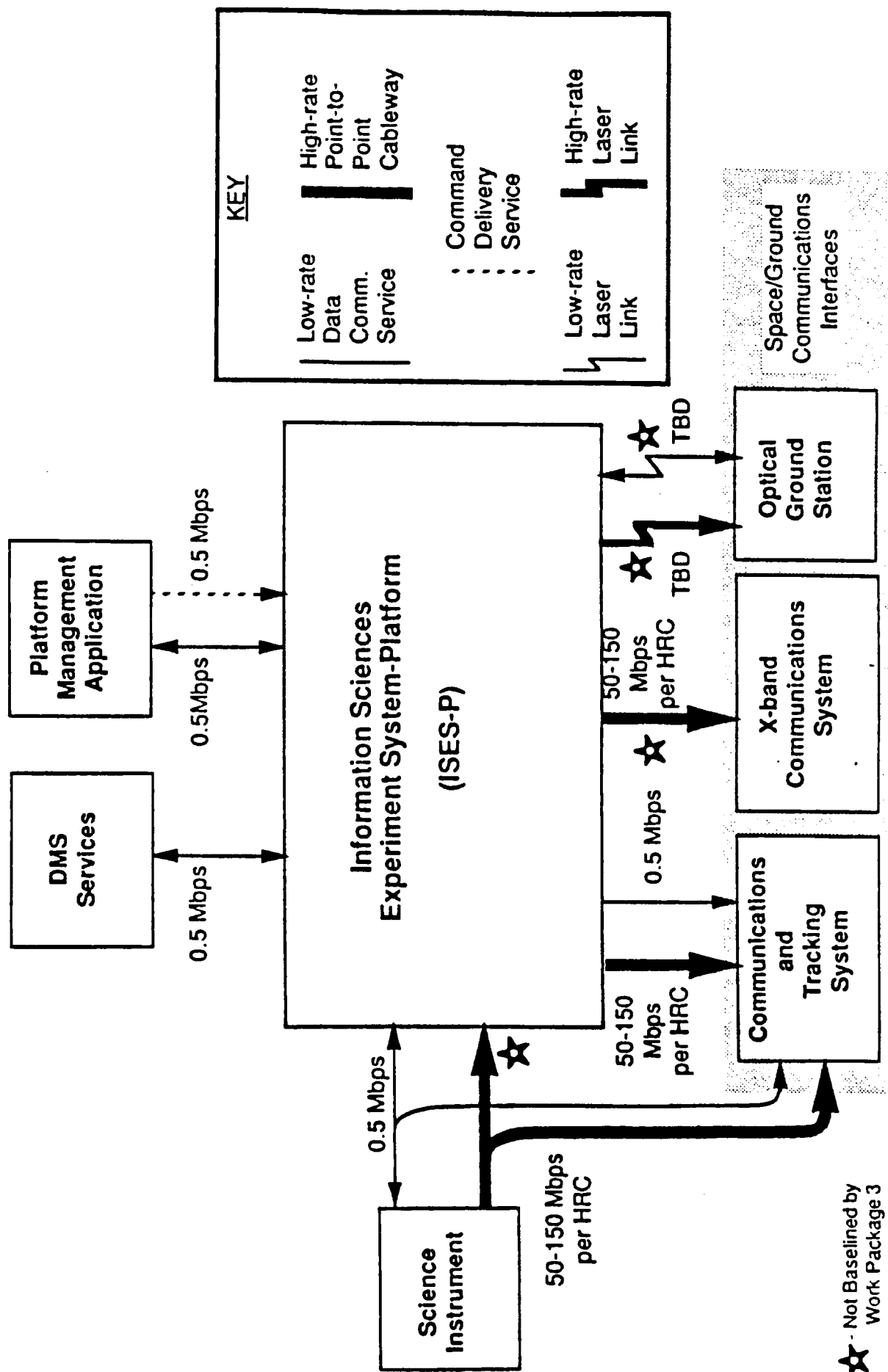
## **THE ISES HOST FACILITY**

- SUPPORT ADVANCED TECHNIQUES AND TECHNOLOGIES
- EXECUTE A WIDE RANGE OF EXPERIMENTAL AND DEVELOPMENTAL ALGORITHMS
- DEMONSTRATE BOTH HIGH-SPEED AND DIRECT-TO-USER LASER COMMUNICATION
- OPERATE ON EITHER POP OR CORE STATION

The second major element supported under this task was interface definition on the Polar Platform and Core Station to both the DMS and experiment data sources. One dimension of ISES is as a "host" to the science experiments and demonstrations. As such the ISES is a piece of "payload-like" hardware that must be compatible with available interfaces and environments. The purpose of the studies was to quantify, as much as possible, the reality of the DMS and its interface and interrelation with users.



# POTENTIAL ON-BOARD DATA COMMUNICATIONS INTERFACES

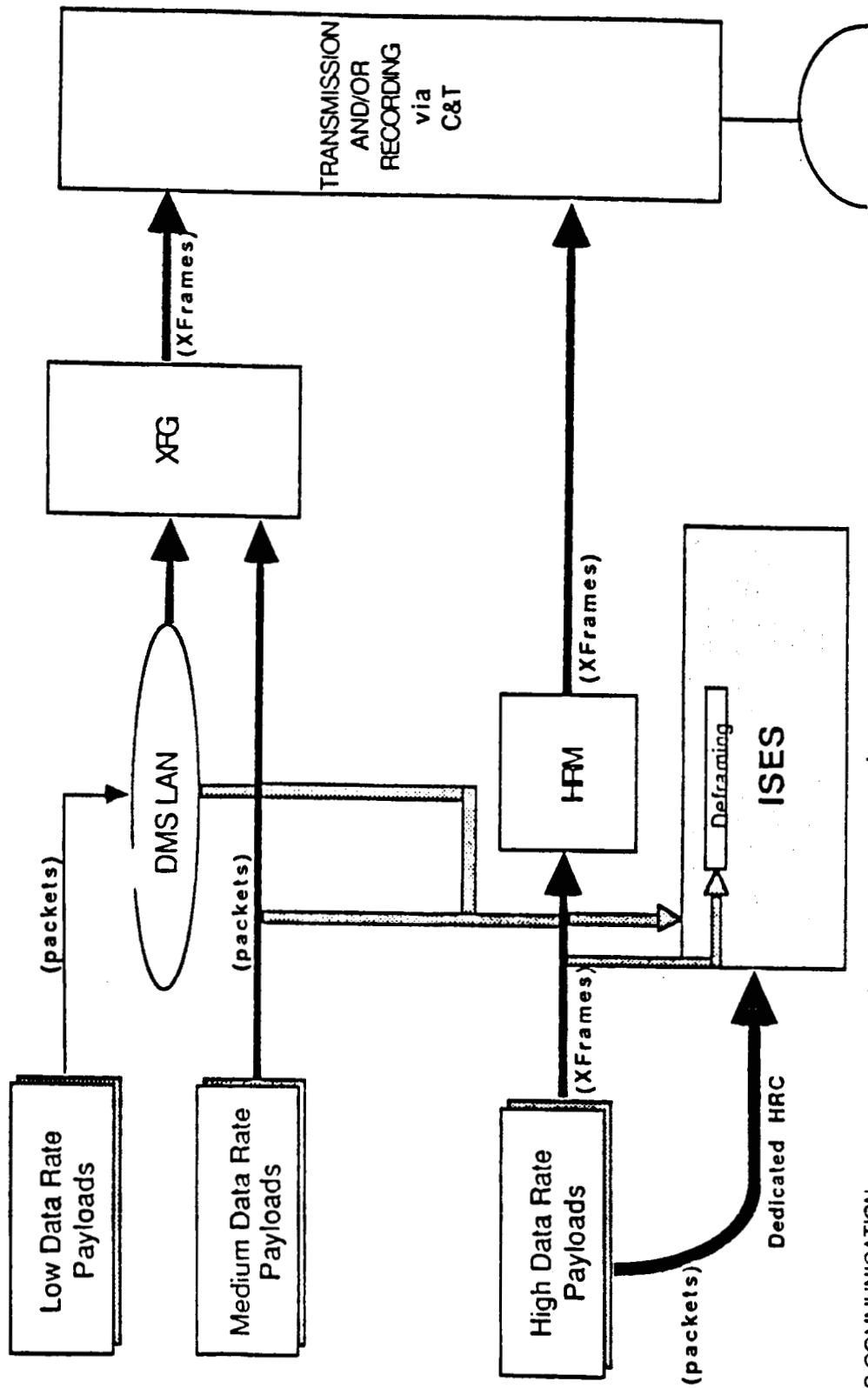


## ON-BOARD DATA COMMUNICATIONS CAPABILITIES

The experiments planned for the ISES facility will require both low-rate and high-rate communications capabilities between ISES and other on-board and ground sources and destinations. The opposing chart illustrate some the planned and potential communications links available to support ISES, with low-rate ( $\leq 0.5$  Mbps) capabilities indicated by narrow lines and high-rate ( $> 0.5$  Mbps) capabilities indicated by bold lines. Low-rate communications services are provided by the Data Management System (DMS) Local Area Network (LAN). High-rate communications are achieved using dedicated point-to-point fiber optic cableways. If necessary, multiple cableways can be installed between a given source and destination to provide the required transmission rate. ISES also has direct interfaces with one or more optical ground stations via the ISES experimental laser communications system. This chart indicates that inter-payload high-rate cableways (HRCs), X-band communications, and optical communications systems have not been baselined by Space Station Freedom Work Package 3.



## Payload to ISES Data Communication



— PL-ISES COMMUNICATION  
— HRC

### Payload to ISES Data Communication

There are several points where ISES could copy payload data. For the low data rate payloads data can be taken from the DMS LAN, and medium rate data can be retrieved from the HRS prior to entering the transfer frame generator. Framing of high rate payload data takes place within the instrument therefore. If high rate payload data is taken from the High Rate Cableway it will be necessary for ISES to incorporate deframing algorithms in its system design. An alternative to this approach would be to use a dedicated path to ISES from the high rate payloads. This would relieve ISES of deframing high rate data.



## Potential ISES Concerns With Platform Data System Concepts

---

- 0.5 Mbps limit per node will necessitate multiple high-rate cableway connections between ISES and other payloads and systems.
- Current policy is that HRCs used only to transport medium and high-rate payload downlink data to C&TS
- Operations management policies and enforcement mechanisms may impact
  - Feasibility of ISES-payload HRC connections
  - Uplinking of data via laser communications system
  - Communications between processes on ISES and other payloads, especially those not running under DMS environment

## **POTENTIAL ISES CONCERNS WITH PLATFORM DATA SYSTEM CONCEPTS**

Based upon the types of experiments to be supported by ISES and the accommodations available through the platform data system, several potential concerns are noted that may influence the ISES concept and/or Space Station Platform requirements. The first concern is that ISES will require the capability to exchange high-rate data with other payloads, but this capability currently is not baselined. The second concern is that the need for the Eos Program to maintain control over on-board operations may affect the perceived feasibility/desirability of permitting direct connections between payloads, e.g., via HRCs. Similarly, the need to maintain control over on-board operations may also affect the perceived feasibility of permitting direct uplinking of data via the ISES laser communications system, and of permitting communications between processes not running under the DMS environment, such as might run on a specialized processor within ISES.

# INFORMATION SCIENCES EXPERIMENT SYSTEM CORE STATION ELEMENT

---

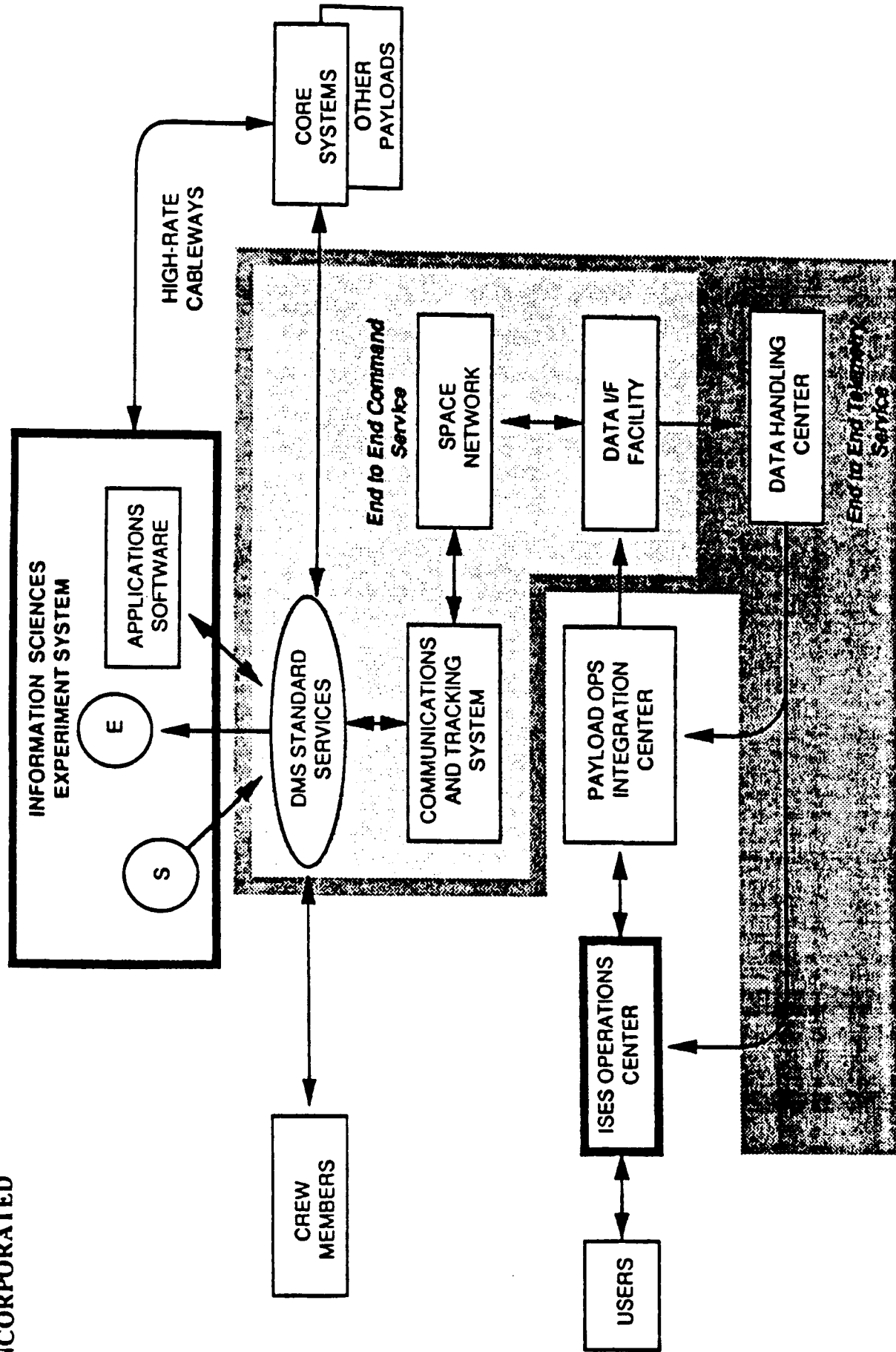
---

- 0 ISES OPERATES THROUGH A DMS WORK STATION PORT.  
IT EAVESDROPS ON DATA AND PRODUCES INFORMATION.
- 0 ISES APPEARS TO THE SPACE STATION DMS AS AN ATTACHED  
PAYLOAD, BUT IN REALITY IS A REMOTE COMPUTING FACILITY  
UNENCUMBERED BY CORE DMS FLIGHT CONSTRAINTS.
- 0 ISES FUNCTIONS IN A SIMILAR FASHION TO POP VERSION, BUT  
SUPPORTS BROADER SCIENCE AND RESEARCH APPLICATIONS  

ATTACHED PAYLOAD DATA	CODE E	SCIENCE GOALS
INSTRUMENTED SPACE STATION	CODE R	TECHNOLOGY GOALS
ADVANCED AUTOMATION	CODE S	A&R GOALS
ADVANCED DATA SYSTEMS TECH	CODE S	SPACE STATION EVOLUTION
- 0 ISES SUPPORTS EXPERIMENTS AND DEMONSTRATIONS OF  
CONCEPTS FOR EVOLUTIONARY SPACE STATION

For the Core Station the ISES concept can be applied to a broader range of science or advanced technology research. The Space Station not only hosts a variety of payloads (Earth science, astrophysics, materials science, biological science) but is also itself a subject of investigation. Research in structural engineering, controls, automation and robotics, and thermal control systems are just a few areas that will receive attention in the development, flight and evolution of the Space Station. An on-board computational resource eavesdropping on data from subsystems offers the opportunity for real-time, parallel, evaluation of competing techniques in the relevant environment. Moreover, the availability of astronaut support permits the evaluation of various hardware options: optical computers, high temperature superconductors, neural net implementations, etc.

# ISES-M OPERATIONAL DATA FLOWS





## ISES-M OPERATIONAL DATA FLOWS

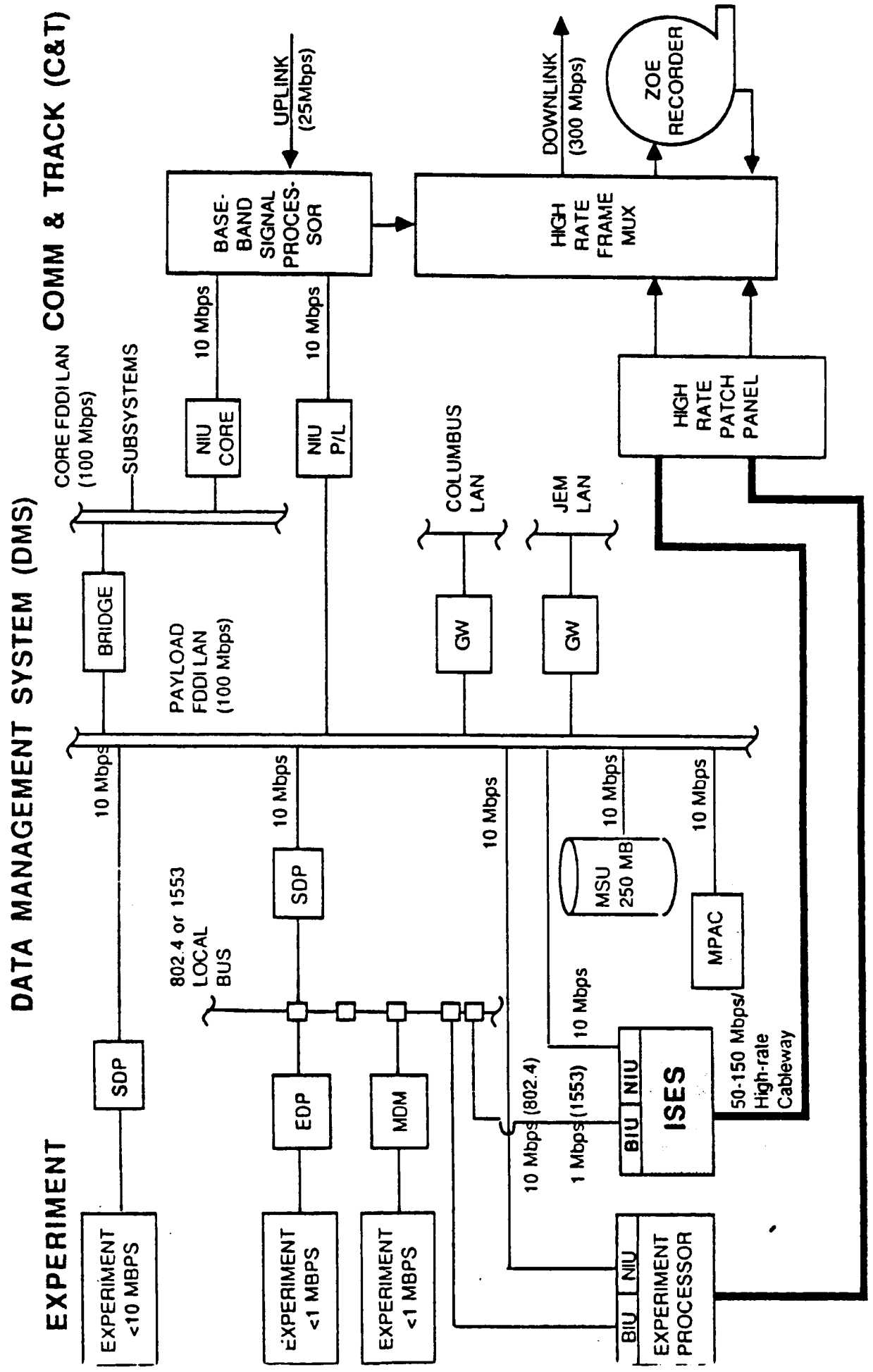
---

ISES-M would use Space Station Information System (SSIS) end-to-end command and telemetry services provided by the DMS, OMS, C&TS, Space Network, Customer Data and Operations System (CDOS), and ISES-M-unique elements, including ISES-M and its ground support system. ISES probably would be managed from a ground-based control center, which would be responsible for maintaining and managing the facility for use by engineers, investigators and technologists. ISES-M operations most likely would be coordinated through the Payload Operations Integration Center (POIC), although a direct interface might exist with the Space Station Control Center to support core system evolution experiments and demonstrations. Other elements of the ISES ground support system are yet to be defined, as are the precise paths by which ISES downlink and uplink data would flow to and from ISES users.

DRAFT



# SPACE STATION DATA SYSTEM



32 US LAB HRCs  
20 EXTERNAL HRCs



## SPACE STATION DATA SYSTEM

The facing page illustrates some of the planned and potential on-board digital data communications links and data interfaces available to support ISES-M. Low-rate (<1 Mbps) communications services are provided by a DMS IEEE 1553 local bus. Medium-rate (<10 Mbps) communications are provided by the DMS 802.4 local buses and the FDDI Local Area Network (LAN). High-rate communications are supported by high-rate cableways (50-150 Mbps per cableway) which are interconnected via a patch panel. By connecting ISES to the high-rate patch panel and FDDI LAN, ISES should have access to all required sources of digital data and associated services, although the adequacy of the communications capacities will depend upon experiment-specific requirements. Access to analog data apparently may not be supported.



INCORPORATED

# SPACE STATION DATA SYSTEM (REPHASED)

## EXPERIMENT

EXPERIMENT  
<10 MBPS

SDP

10 Mbps

CORE/PAYLOAD LAN  
(100 Mbps)

1553  
LOCAL  
BUS

10 Mbps

SDP

10 Mbps

NIU  
P/L

10 Mbps

BASE-  
BAND  
SIGNAL  
PROCES-  
SOR

UPLINK  
(25M)

EXPERIMENT  
<1 MBPS

MDM

BIU NIU  
EXPERIMENT  
PROCESSOR

BIU NIU  
ISES

1 Mbps

10 Mbps

MSU  
250MB

10 Mbps

MPAC

GW

COLUMBUS  
LAN

GW

JEM  
LAN

100 Mbps  
(FDDI aggregate  
capacity)

HIGH  
RATE  
PATCH  
PANEL

HIGH  
RATE  
FRAME  
MUX

DOWNLINK  
(50 M)

ZOE  
RECORDER

## DATA MANAGEMENT SYSTEM (DMS)

## COMM & TRACK (C&T)

• Possible in principle, but only 2 HRCs planned



## SPACE STATION DATA SYSTEM (REPHASED)

The facing page illustrates some of the most significant impacts of the Space Station Freedom rephasing/descoping with respect to providing ISES with access to other on-board sources and destinations of data. Rather than providing two separate FDDI LANs, one each for core systems and payloads, it has been recommended that NASA provide only a single FDDI LAN, to be shared between payloads and core systems. In addition, the 802.4 local bus is to be eliminated in favor of the simpler but less powerful 1553 local bus. Despite these changes, however, ISES could still access payload and core system data via the FDDI network at 10 Mbps through a Network Interface Unit, via high-rate cableways (HRCs) at 50-150 Mbps per HRC. It is possible that the 1553 local bus could augment the other two media, but it is not adequate by itself to support ISES.

## CONCLUSION

- SEVERAL PRE-PHASE A STUDIES HAVE BEEN COMPLETED FOR THE INFORMATION SCIENCES EXPERIMENT SYSTEM (ISES).
- FOR THE POLAR PLATFORM, A SET OF HIGH PAYOFF EXPERIMENT AREAS HAVE BEEN DEFINED THAT WILL INITIALIZE FURTHER DEFINITION ACTIVITIES.
- INTERFACE ACCOMMODATION STUDIES HAVE BEEN COMPLETED FOR BY THE POLAR PLATFORM AND CORE STATION.
- TWO ADDITIONAL STUDIES, ONE ON EOS SCIENCE INSTRUMENT-TO-INSTRUMENT INFORMATION EXCHANGE AND THE OTHER ON CORE STATION EXPERIMENT IDENTIFICATION, WILL BE COMPLETED SHORTLY.
- THE RESULTS OF THE CURRENT STUDIES HAVE CONFIRMED THE VIABILITY OF THE ISES CONCEPT FOR RAPID INSERTION OF ADVANCED INFORMATION SYSTEM TECHNOLOGIES INTO SPACE FLIGHT APPLICATIONS.

**AUTONOMOUS CONTROL  
(FORMERLY LAUNCH OPERATIONS)**

**THE KNOWLEDGE-BASED AUTONOMOUS TEST ENGINEER (KATE)  
AT  
JOHN F. KENNEDY SPACE CENTER (KSC)**

**NASA, KSC A.I. LABORATORY  
MCDONNELL DOUGLAS SPACE SYSTEMS CO. - KSC**

**FEBRUARY 8, 1990**

**TASK MANAGER: DAVE SIAS, NASA-KSC**

**PROJECT TEAM:**

BARBARA BROWN	-	NASA-KSC TECHNICAL LEAD
MEL MEEKINS	-	MDSSC-KSC PROJECT ENGINEER
JOHN ROLLINS	-	MDSSC-KSC TECHNICAL LEAD
RAY HO	-	MDSSC-KSC TECHNICAL TEAM MEMBER
MIKE MYJAK	-	MDSSC-KSC TECHNICAL TEAM MEMBER

## **PROJECT MOTIVATION**

- o SPACE STATION FREEDOM NEED FOR AUTONOMOUS ON-BOARD SOFTWARE SYSTEMS (SPECIFICALLY A.I. SOLUTIONS)
- o NEED FOR EVALUATION OF INTEL 80386 PROCESSOR AS A DELIVERY VEHICLE FOR A.I. SYSTEMS
- o NEED FOR EVALUATION OF THE USE OF ADA PROGRAMMING LANGUAGE TO IMPLEMENT A.I. SYSTEMS IN A 80386 ENVIRONMENT
- o NEED TO TEST A.I. SYSTEMS IMPLEMENTED ON A CONVENTIONAL COMPUTER ARCHITECTURE

## **PROJECT HISTORY**

- o **KATE HISTORY**
  - **BASED ON LES -- LIQUID OXYGEN EXPERT SYSTEM**
  - **LES ORIGINALLY DEVELOPED BY MITRE, INC./NASA-KSC**
  - **KATE IS AN EXPANSION OF LES THAT INCLUDES CONTROL FEATURES**
- o **KATE IS USED TO MONITOR, DIAGNOSE, AND CONTROL ELECTRO-MECHANICAL DEVICES USING A MODEL-BASED REASONING APPROACH**
- o **CURRENT KATE DEVELOPMENT BEING DONE BY NASA/BOEING AT KSC**
- o **NASA/BOEING-KSC IS CURRENTLY PORTING KATE FROM A SYMBOLICS LISP MACHINE TO A TEXAS INSTRUMENTS LISP MACHINE (T.I. EXPLORER)**

## **OBJECTIVES**

- **EVALUATE THE POTENTIAL USE OF A.I. TECHNOLOGY TO DELIVER AUTONOMOUS SOFTWARE SYSTEMS FOR SPACE STATION APPLICATIONS**
- **DETERMINE THE FEASIBILITY OF USING ADA TO DELIVER A.I. SOFTWARE SYSTEMS**
- **COMPARE THE PERFORMANCE OF A VERSION OF KATE DEVELOPED IN LISP TO A VERSION DEVELOPED IN ADA**
- **COMPARE THE PERFORMANCE OF THE CONVENTIONAL 80386 PROCESSOR TO A SPECIAL-PURPOSE SYMBOLIC PROCESSOR AS A RUN-TIME PLATFORM FOR A.I. SYSTEMS**
- **PROVIDE INSIGHT INTO THE DEVELOPMENT OF AN A.I. SOFTWARE SYSTEM IN ADA ON A CONVENTIONAL 80386 PROCESSOR**

## **APPROACH**

- o **USE THE NASA/BOEING-KSC T.I. VERSION OF KATE AS A BASELINE. THIS VERSION WILL BE INTEGRATED AS A DISPLAY PROCESSOR IN THE GCS NETWORK**
- o **ESTABLISH PERFORMANCE CRITERIA AND TEST PLANS TO GUIDE ACCURATE MEASUREMENT AND COMPARISON OF THE PERFORMANCE OF EACH VERSION OF KATE**
- o **PORT LISP VERSION OF KATE IMPLEMENTED ON A T.I. EXPLORER TO THE 80386 PROCESSOR**
- o **INTEGRATE THE 80386 PLATFORM AS A DISPLAY PROCESSOR IN THE GENERIC CHECKOUT SYSTEM (GCS) NETWORK, PROVIDING ACCESS TO THE RED WAGON TEST ARTICLE**

## **APPROACH**

**(CONT)**

- o **CONVERT 80386 LISP VERSION OF KATE TO AN 80386 ADA VERSION**
- o **EVALUATE THE PERFORMANCE OF EACH VERSION OF KATE ON EACH PLATFORM USING THE SAME TEST-BED (RED WAGON) AND PERFORMANCE TEST PLANS**
- o **COMPARE CONVENTIONAL PLATFORMS AND LANGUAGES TO SPECIAL-PURPOSE PLATFORMS AND ENVIRONMENTS FOR THE DEVELOPMENT AND DELIVERY OF A.I. SOFTWARE SYSTEMS**

## **ACCOMPLISHMENTS**

- o **EQUIPMENT REQUIREMENTS ANALYSIS**
  - **INVESTIGATED SPACE STATION HARDWARE AND SOFTWARE STANDARDS**
  - **DEFINED GCS NETWORK HARDWARE AND SOFTWARE REQUIREMENTS**
  - **DEFINED REQUIREMENTS FOR KATE TO BE A GCS DISPLAY PROCESSOR**
  - **DEFINED KATE 80386 VERSIONS' HARDWARE AND SOFTWARE REQUIREMENTS**
- o **PROCUREMENT**
  - **DEVELOPED ADP ACQUISITION PLAN FOR 80386 PLATFORM**
  - **PREPARED HARDWARE AND SOFTWARE SPECIFICATIONS FOR 80386 PLATFORM**
  - **SENT REQUEST FOR QUOTES (RFQ) TO QUALIFIED HARDWARE AND SOFTWARE VENDORS**
  - **EVALUATED BIDS ON TECHNICAL MERIT**
  - **PROCUREMENT OF ONE (1) 80386 PLATFORM IS AWAITING FY90 FUNDS**
- o **PERFORMED TECHNICAL EVALUATIONS**
  - **INVESTIGATED LISP TO ADA TRANSLATORS**
  - **EVALUATED USER INTERFACE TOOLS FOR DEVELOPING KATE USER INTERFACE ON THE 80386 MACHINE**
- o **DEVELOPED DOCUMENT GIVING OVERVIEW OF PERFORMANCE CRITERIA AND PERFORMANCE TEST PLAN**

## **FY90 PLANS**

- o **DEVELOP KATE FUNCTIONAL REQUIREMENTS DOCUMENT**
- o **COMPLETE PROCUREMENT OF ONE (1) 80386 WORKSTATION**
- o **DEVELOP A PERFORMANCE TEST PLAN AND PERFORMANCE TEST PROCEDURES**
- o **BEGIN PORTING THE T.I. VERSION OF KATE TO THE 80386 IN LISP**
  - **FIRST PHASE IS A STAND-ALONE VERSION**
  - **IDENTIFY SYSTEM-DEPENDENT LISP CODE**
  - **RE-WRITE SYSTEM-DEPENDENT CODE (MAINLY USER INTERFACE)**
- o **MEASURE THE PERFORMANCE OF THE T.I. VERSION OF KATE**

## **FY90 PLANS**

**(CONT)**

- o BEGIN MEASURING THE PERFORMANCE OF THE 80386 LISP VERSION OF KATE**
- o BEGIN DEVELOPMENT OF A DETAILED DESIGN DOCUMENT THAT WILL GUIDE THE EVENTUAL RE-CODING OF KATE INTO ADA**
- o GIVE FOLLOW-ON DEMONSTRATIONS OF T.I. VERSION OF KATE**
- o INVESTIGATE SOFTWARE SUPPORT ENVIRONMENT (SSE) SOFTWARE DEVELOPMENT TOOLS**

## Facing Page for PROJECT MOTIVATION

Advanced technologies play an integral part in the development of the Space Station Freedom (SSF) on-board systems. Limited manpower aboard the Space Station necessitates the automation of many tasks that require decision-making capabilities. Intelligent, autonomous software systems provide the core for automating these tasks. Congress has recognized this need for autonomous software systems and is funding their research and development.

Artificial Intelligence (A.I.) software has been an effective vehicle for delivering intelligent, autonomous software systems. A.I. technology allows one to model a problem in a logical, well-structured, "human-like" fashion. This economy of representation, coupled with powerful problem-solving methods, enables one to tackle complex, heavily constrained problems. Often, an A.I. approach can produce an acceptable solution to a non-deterministic or previously intractable problem.

The Space Station Program Office (SSPO) has managed the development of documents that specify Space Station computing standards. These standards specify that on-board applications requiring Data Management Systems (DMS) services be delivered on Intel 80386 platforms and that the Software Support Environment (SSE) for these applications be written in the Ada programming language. Therefore, all on-board A.I. software requiring DMS services should be implemented in Ada on an 80386 machine.

Historically, A.I. software has resided on special purpose processors and has been written in symbolic languages (e.g. - LISP). This has allowed rapid prototyping but has also produced very resource-exhaustive software. Consequently, A.I. software has rarely been implemented in a real-time, closed-loop control environment or in Ada.

It is important to determine the feasibility of delivering A.I. systems in the Space Station standard computing environment (80386 and Ada) and thus verify the suitability of using A.I. technology to implement on-board autonomous software systems.

The Knowledge-Based Autonomous Test Engineer (KATE) is an A.I. diagnostics system that is developed in LISP on a special purpose Texas Instruments Explorer II (T.I.) machine. Its model-based reasoning approach shows potential benefit for Space Station applications. This makes KATE a good test-bed for evaluating the conventional Space Station computing technology as a platform for A.I. systems.

## **Facing Page for PROJECT HISTORY**

KATE evolved from a research product, the Liquid oxygen Expert System (LES), jointly developed by Mitre, INC and NASA-KSC. LES was developed to monitor and diagnose a liquid oxygen system at the Kennedy Space Center. Along with a number of utilities, KATE adds to LES the ability to exercise control over the system being monitored and diagnosed.

KATE is a generic tool which autonomously monitors, diagnoses, and controls electro-mechanical devices. It employs a model-based reasoning approach in representing and solving a diagnostics problem. KATE is generic in the sense that it uses the same algorithms regardless of the specific system being monitored, diagnosed and controlled. Only the model of the system (device characteristics, relationships between devices, etc.) is specific to the application. This model is developed by a domain expert and stored in KATE's knowledge-base.

KATE is still under development at NASA-KSC in the A.I. lab where enhancements are being made by a NASA/Boeing team. Most recently, Boeing is porting KATE from a Symbolics LISP machine to a Texas Instruments Explorer LISP machine.

## **Facing Page for OBJECTIVES**

A main objective of the current work is to determine if A.I. software systems can meet the requirements for on-board Space Station computer applications.

KATE, like much A.I. software, was developed in LISP. Currently, only ADA is approved for on-board applications. The KATE/Ada project has an objective of seeing if an A.I. system can be successfully implemented in Ada.

As part of the feasibility study, a performance comparison will be made between the LISP and ADA versions of the KATE software.

Due to the choice of LISP for much A.I. software, many systems are developed on special purpose LISP machines. It is an objective of this project to port KATE to a conventional 80386 platform in LISP and to measure the performance against the original version of the software implemented on a LISP machine.

Finally, it is an objective of the current work to integrate all of the issues. This will be accomplished by measuring the performance of a version of KATE written in Ada for the 80386 platform. This version's performance and ease of development will then be compared to that of the other versions (i.e. - the T.I. LISP version and the 80386 LISP version). This KATE/Ada work will yield insight into the use of model-based A.I. technology within a conventional computing environment.

## **Facing Page For GCS LAB PICTURE - KSC-389C-4523.01**

The KATE system will be imbedded in the GCS test-bed network, where it will control the Red Wagon test article. This enables KATE to be tested under real-time, closed-loop control conditions.

The GCS test-bed is located in the Engineering Development Laboratory (EDL) at KSC. Several real-time control applications run from the GCS network. They collect data from Data Acquisition Modules (DAM), control test articles (Red and Blue Wagons), and display the results on Display Processors (DP).

## **Facing Page for GCS NETWORK PICTURE - KSC-389C-4523.10**

The GCS network consists of Display Processors (DP), Application Processors (AP), Data Acquisition Modules (DAM), Data Acquisition Processors (DAP), Test Articles, and Ethernet networks that allow communications among the equipment. An application resides on an AP. The application may poll the DAMs and display information on the DPs. The application may also control the test articles through output hardware.

Because of security issues, KATE will be implemented as a DP on the GCS network. KATE will receive information from the Red Wagon via a DAM and will display its activity and results via its console. KATE will also control the Red Wagon hardware by using redundant components to bypass faults that it detects.

## **Facing Page for RED WAGON TEST ARTICLE PICTURE - KSC-389C-4523.07**

The Red Wagon is a prototype of the shuttle tanking system. It simulates the filling and draining of the shuttle fuel tank. The Red Wagon has several redundant paths for controlling the tank. The hardware components on a given fueling path consist of pumps, solenoid valves, and piping. Flow meters and pressure sensors give measurements to the control system.

KATE receives measurements from the sensors and can control the flow rate of pumps and can command valves to change their state. In the event of a failure, KATE may establish an alternate fueling path by controlling the proper pumps and valves.

Today, there are no 80386 processors on the GCS network. It is important to verify that an 80386 will work properly as a subsystem in the GCS network. Before porting KATE from the T.I. to the 80386, MDSSC will verify the 80386 processor's ability to operate as a DP on the GCS network.

## Facing Page for APPROACH

Currently, NASA/Boeing-KSC is porting KATE from a Symbolics LISP machine to a T.I. Explorer. They will connect the T.I. processor to the GCS network and demonstrate KATE diagnosing and controlling the Red Wagon hardware. McDonnell Douglas Space Systems Company at Kennedy Space Center (MDSSC-KSC) will then use the T.I. version of KATE as its baseline system.

Performance measurement criteria and associated performance test plans must be established. These performance plans will ensure that the performance of each of the versions of KATE is accurately compared. The performance criteria and test plans will specify the parameters to be measured, the components of KATE to monitor, the procedures for measuring performance, and the analysis to be performed on the resulting performance data. The performance criteria and test plans will be applied uniformly against all versions of KATE to ensure accurate comparisons.

Following the T.I. KATE/Red Wagon demonstrations, the T.I. version of KATE will be ported to the 80386 processor. Much of the T.I. version of KATE is written in Common LISP. Porting this part of KATE should be straight-forward. Two significant modules of KATE, the user interface and the network interface, are not written in Common LISP. This necessitates completely re-writing those modules to run on the 80386 processor under UNIX. Software tools that ensure portability among a large group of processors will be used to re-write the user interface module. The network interface module will be written in the language C as specified by the GCS network protocol.

The ported LISP version of KATE will be imbedded in the GCS network as a DP. KATE will be demonstrated monitoring, diagnosing, and controlling the Red Wagon test article. This 80386 demonstration will follow same demonstration plan used for the T.I. version of KATE.

## Facing Page for APPROACH (Cont.)

KATE will next be re-written from the LISP version into Ada on the 80386. This will be accomplished by creating a Detailed Design Document. The Detailed Design Document will describe KATE's design in such a manner that an Ada programmer may code directly from it. Many technical issues will arise during this stage of the project. KATE is written using an object-oriented methodology. The best way to represent that methodology in Ada is currently unclear. Also, KATE employs LISP features that are not present in Ada and, most certainly, there is not a one-to-one functional mapping between LISP and Ada. The Detailed Design Document will resolve these issues. Once KATE has been re-written in Ada, it will again be demonstrated as a GCS sub-system against the Red Wagon.

The performance of each version of KATE (T.I. LISP, 80386 LISP, 80386 Ada) will be measured and documented. The same test plan will be used to test each version's performance. These performance tests will give insight into how well model-based expert systems perform in conventional computing environments as opposed to special-purpose computing environments.

A final report discussing the results as they pertain to the project's objectives will be published. The final report will evaluate the advantages and disadvantages of developing A.I. systems in Ada compared to LISP. It will analyze the performance of A.I. systems running in the standard Space Station computing environment (80386 and Ada) and in special-purpose computing environments (LISP and symbolic processors). The final report will contain recommendations about how to improve the performance of conventional environment-based A.I. systems and will discuss tradeoffs imposed by a chosen implementation environment.

## Facing Page for ACCOMPLISHMENTS

For the KATE software to be tested in an environment which supports the objectives of this project, the KATE system must be integrated with other software and be able to access real hardware. The Generic Checkout System (GCS) network, which is connected to the Red Wagon Test-Bed, is one such environment. An equipment requirements analysis was performed to determine the hardware, software, and networking requirements to allow KATE to be integrated with GCS and therefore to have access to the Red Wagon hardware. The Space Station standards to which the KATE software and hardware environment must conform were also investigated. Finally, the hardware and software needed to support the KATE system itself was identified.

An ADP acquisition plan was developed to guide purchasing an 80386 development workstation. The ADP acquisition plan specified the equipment to be procured, the phased plan for procuring the equipment, and the method of procurement. An equipment technical specification was prepared that stated precisely the technical qualities that the purchased workstation software and hardware must exhibit. Vendor quotes were requested and received. The quotes were evaluated based on their adherence to the technical specifications. Procurement of one (1) 80386 workstation has been put on hold until the obligated FY90 funds arrive.

During the equipment requirements analysis, technical evaluations were conducted to investigate different alternatives to meet program objectives. First, LISP to ADA translators were investigated to determine if they could be used during the re-coding of KATE into ADA. Although more investigation is needed, early results show an absence of usable translators. Investigation into software tools to develop the 80386 KATE user interface were explored. The T.I. version of the user interface was written in system dependent code. This code will not port to the 80386 machine and will have to be re-written. The current research has identified two potential choices for non-system dependent user interface development tools for the 80386. These are MIT X11-Windows and associated toolkits and Data Views.

To indicate the direction that MDSSC proposes for measuring KATE's performance consistently across all KATE versions, MDSSC developed a document that outlines the performance criteria and test plan. This overview document set the guidelines and approach for all performance testing and analysis. Future detailed performance testing documents will expand on this overview document.

## Facing Page for FY90 PLANS

Originally, the FY90 KATE/Ada MDSSC-KSC manpower level was six people. The FY90 budget announcement has reduced that to three people. The KATE/Ada work for FY90 is being re-scoped to match this funding level. A Functional Requirements Document is being created to facilitate this process. The document will also serve to guide all future KATE/Ada work to ensure that it meets the project requirements. The Functional Requirements Document contains the objectives, scope, constraints, and resource requirements of the project.

One (1) 80386 development workstation with software will be procured in FY90.

To meet the objectives of the KATE/Ada project, the performance of each version of KATE must be measured. Criteria and methods by which to measure KATE's performance have already been outlined. Now, a performance test plan that guides the performance measurements must be established so that each version of KATE is measured the same way. The test plan will establish the goals of the performance tests, the parameters to be measured, the components of KATE that will be measured, and how those measurements will be accomplished. The test plan will guide the development of performance test procedures. The test procedures will be designed to be minimally modified for each subsequent performance test.

The T.I. version of KATE will be ported to a LISP version on the 80386 platform.

The first phase of this port is to create a stand-alone version of KATE in LISP on the 80386 platform. (Stand-alone means not integrated in the GCS network). To enable valid comparisons, the 80386 LISP version will be as identical as possible to the T.I. version.

The first step in performing the port is to identify the KATE modules that have been written to run specifically on the Texas Instruments hardware platform. Before KATE can be ported, these hardware-dependent modules will have to be re-written either with user interface development tools or in pure Common LISP on the 80386 workstation.

Once all hardware-dependent code has been identified and re-written, KATE will be ported to the 80386 machine. This version of KATE will be written in ANSI standard Common LISP with machine-independent user interface software. This makes KATE portable to any 80386 processor that has a sufficient amount of computing resources.

The performance of the T.I. version of KATE will be measured using the test procedures. The results of the measurements will be documented.

## **Facing Page for FY90 PLANS (Cont.)**

The procedures previously developed to measure the T.I. version of KATE will be modified to measure the performance of the 80386 LISP version. All effort will be made to ensure unbiased comparison of the two versions' performance. The results of the measurements will be documented. (This task is actually not scheduled until November, GFY91.)

A Detailed Design Document will be started based on the 80386 LISP version. The Detailed Design Document will direct the re-write of KATE into Ada. The document will be slanted as little as possible towards Ada. The document will describe the functions, structures, and methods of KATE in detail. With minimal effort, one may then use the document to create a version of KATE in any language powerful enough to represent KATE's design concepts.

Once KATE is delivered on the T.I. platform as a sub-system in the GCS network, MDSSC will demonstrate its capabilities to interested parties upon request. These demonstrations will give an in-depth view into the capabilities of KATE as a real-time diagnostics and control system.

The Software Support Environment (SSE) Tools is a standard set of software development tools to aid software developers with life-cycle software management. The applicability of using SSE Tools to KATE's model-based reasoning approach will be investigated. SSE Tools may be used to create the design documents for re-writing KATE into Ada.

**SESSION X**  
**TELEROBOTICS TECHNOLOGY AND APPLICATIONS**

**Session Chair:**  
**Mr. Wayne Zimmerman**  
**Office of Space Station**

Task Title : **TELEROBOTIC SYSTEM TECHNOLOGY**

WBS Element: **Robotic System Technology**

Participants: **P. Backes, T. Lee, K. Tso**

Task Manager: **S. Hayati**

**Final Report, FY '89 (draft)**

**Jet Propulsion Laboratory, California Institute of Technology**

# TELEROBOTIC SYSTEM TECHNOLOGY

## Content

- Goals and Objectives
- Approach
- Architecture
- Teleoperation
- Autonomous Control
- Traded Control
- Shared Control
- Supervisory Control
- Accomplishments
- FY 90 plans

# TELEROBOTIC SYSTEM TECHNOLOGY

## Goals and Objectives

- Develop traded and shared control technology to enhance baseline design capabilities. Specific goals are:
  - Increase productivity
  - Enhance operator's ability to perform tasks
  - Perform tasks from an Earth-based station
  - Allow human to do what human does best and autonomy to do what autonomy does best
  - Design a control system that is robust to transmission time-delays
  - Allow the operator to build new functionalities based on the available tools on-line

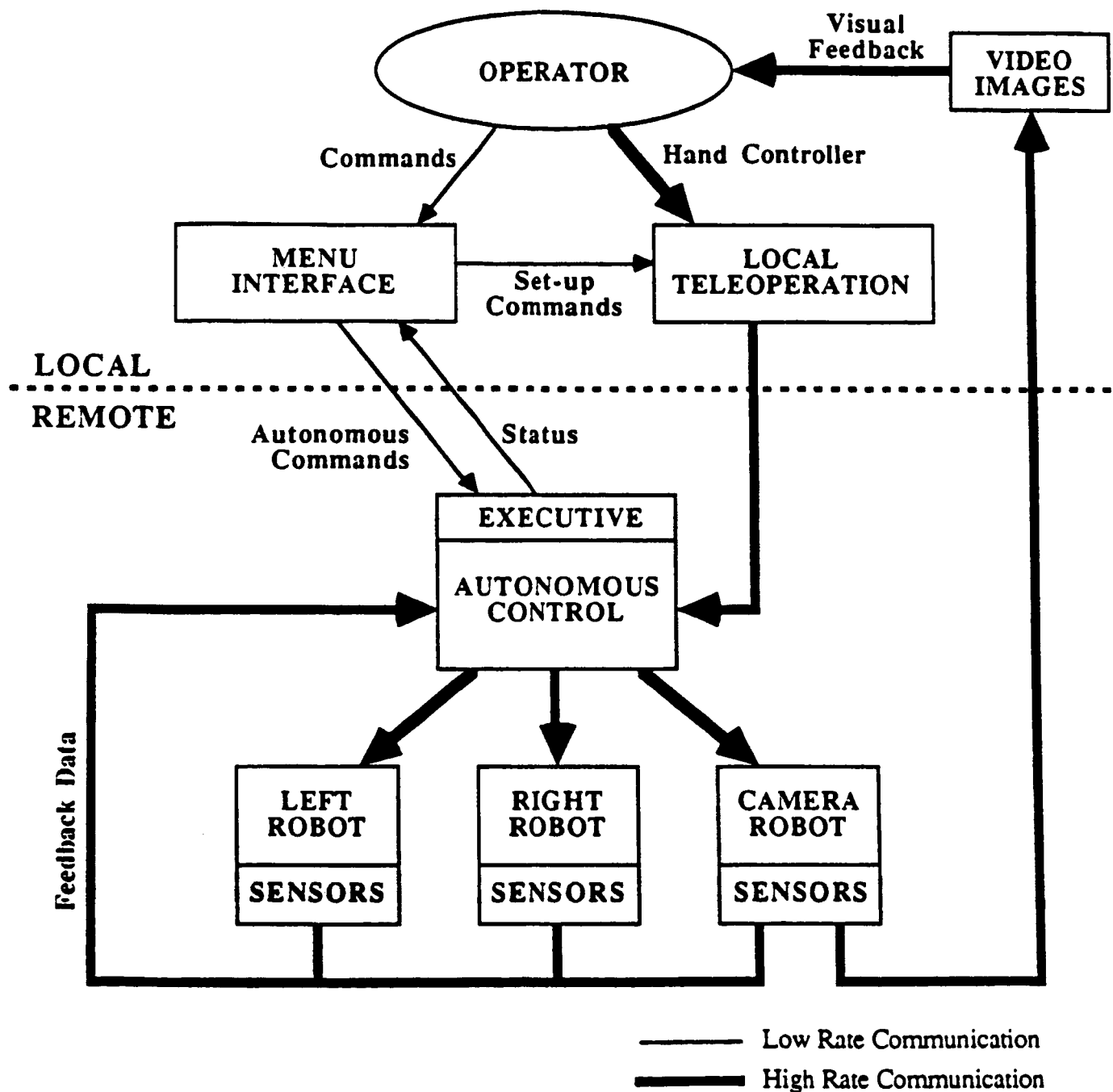
# TELEROBOTIC SYSTEM TECHNOLOGY

## Approach

1. **Traded Control:** Transition between teleoperation and autonomous control easily and frequently
2. **Shared Control:** Simultaneous autonomous control and teleoperation
3. **Supervisory Control:** Operator determines how a task will be performed and uses available software tools to execute a task interactively

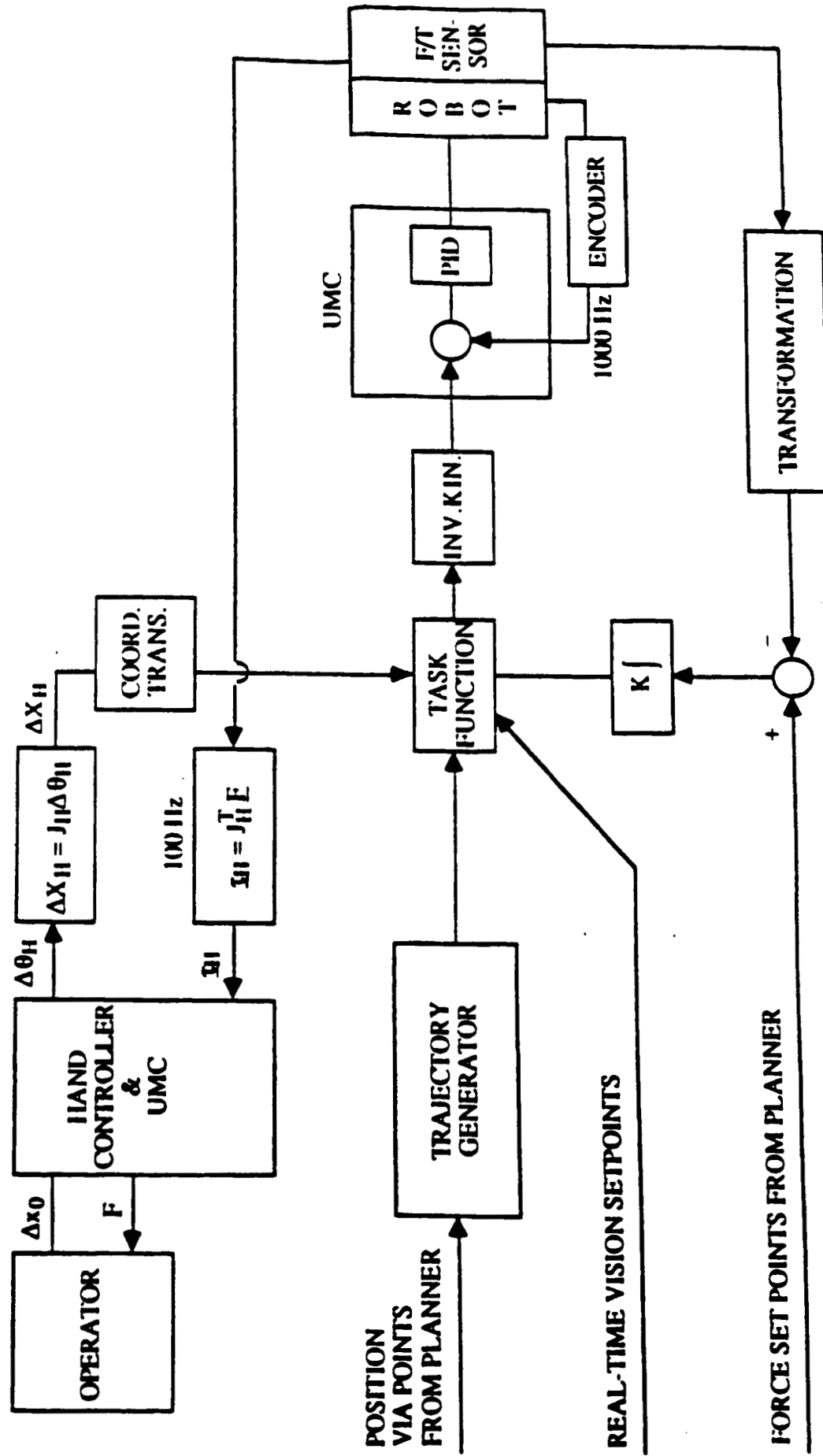
# TELEROBOTIC SYSTEM TECHNOLOGY

## Functional Architecture



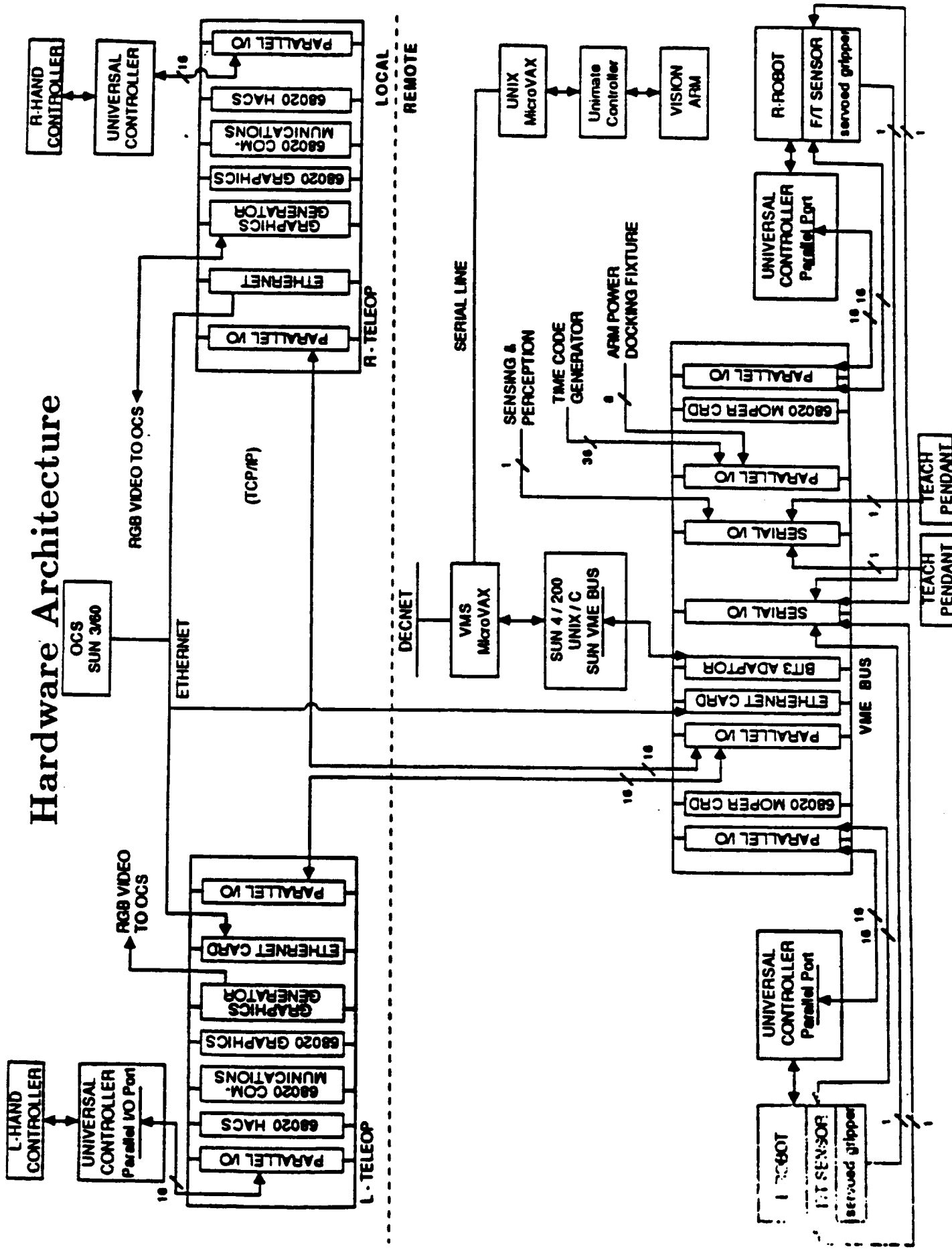
# TELEROBOTIC SYSTEM TECHNOLOGY

## Control Block Diagram



# TELÉROBOTIC SYSTEM TECHNOLOGY

## Hardware Architecture



# TELEROBOTIC SYSTEM TECHNOLOGY

## Teleoperation Capability

- Developed non-replica master slave dual-arm force reflecting teleoperation
- Modes of teleoperation:
  - Joint space
  - Cartesian tool space
  - Cartesian world space
- User menu for setting up teleoperation parameters
- Automatic mode switch at joint limits, joint singularities
- Voice commanding and feedback

## Teleoperation User Interface

Teleoperation Control			
Arm: <input type="checkbox"/> Left <input type="checkbox"/> Right <input checked="" type="checkbox"/> Dual Hand: <input type="checkbox"/> Left <input type="checkbox"/> Right <input checked="" type="checkbox"/> Dual		<input type="button" value="RUN"/> <input type="button" value="QUIT"/> <input type="button" value="NORMAL MENU"/>	
<b>Command to the Left Arm</b>		<b>Command to the Right Arm</b>	
Mode: <input type="checkbox"/> World <input checked="" type="checkbox"/> Tool <input type="checkbox"/> Joint Tool: ORL No Tool Gripper: TRI Gripper <input type="checkbox"/> Close Delta (mm) = 0 Speed (mm/s) = 15 Accel (mm/s/s) = New Tool Force (N) = 20		Mode: <input type="checkbox"/> World <input checked="" type="checkbox"/> Tool <input type="checkbox"/> Joint Tool: TRI Gripper Gripper: <input type="checkbox"/> Idle <input type="checkbox"/> Open <input checked="" type="checkbox"/> Close Delta (mm) = 5 Speed (mm/s) = 15 Accel (mm/s/s) = 10 Force (N) = 20	
<b>Cartesian Scale      Force Gain (Dual)</b> Tx = 1.00      Fx = 2.00 Ty = 1.00      Fy = 2.00 Tz = 1.00      Fz = 2.00 Rx = 1.00      Tx = 0.70 Ry = 1.00      Ty = 0.70 Rz = 1.00      Tz = 0.70		<b>Cartesian Scale      Force Gain (Dual)</b> Tx = 1.00      Fx = 2.00 Ty = 1.00      Fy = 2.00 Tz = 1.00      Fz = 2.00 Rx = 1.00      Tx = 0.70 Ry = 1.00      Ty = 0.70 Rz = 1.00      Tz = 0.70	
<b>System Message: System Normal</b>			
<b>Status of the Left Arm</b>		<b>Status of the Right Arm</b>	
Joint 1 [0] -155 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 155 Joint 2 [0] -210 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 30 Joint 3 [0] -47 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 227 Joint 4 [0] -115 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 185 Joint 5 [0] -95 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 95 Joint 6 [0] -174 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 174  Fx [0] -150 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 150 Fy [0] -150 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 150 Fz [0] -150 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 150 Tx [0] -999 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 999 Ty [0] -999 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 999 Tz [0] -999 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 999  Gripper [0] 0 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 00		Joint 1 [0] -155 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 155 Joint 2 [0] -210 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 30 Joint 3 [0] -47 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 227 Joint 4 [0] -115 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 185 Joint 5 [0] -95 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 95 Joint 6 [0] -174 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 174  Fx [0] -150 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 150 Fy [0] -150 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 150 Fz [0] -150 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 150 Tx [0] -999 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 999 Ty [0] -999 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 999 Tz [0] -999 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 999  Gripper [0] 0 <div style="display: inline-block; width: 100px; height: 15px; background: linear-gradient(to right, black 50%, white 50%);"></div> 00	
Hand Connection: <input type="checkbox"/> ON <input checked="" type="checkbox"/> OFF		Hand Connection: <input type="checkbox"/> ON <input checked="" type="checkbox"/> OFF	
Force Feedback: <input checked="" type="checkbox"/> OFF <input type="checkbox"/> ON		Force Feedback: <input type="checkbox"/> OFF <input checked="" type="checkbox"/> ON	

# TELEROBOTIC SYSTEM TECHNOLOGY

## Autonomous Control Capability

- Developed autonomous control primitives
  - Generalized Compliant Motion (Hinge, slide, insert, ...)
  - Joint Guarded motion
  - Cartesian Guarded motion
  - Move to Touch
- Autonomous primitives provide tools for autonomous task execution

# TELEROBOTIC SYSTEM TECHNOLOGY

## Traded Control Capability

Traded control: tradeoff between autonomous and teleoperation control during task execution

- Execute part of task with teleoperation, e.g., positioning, and other part with autonomous control, e.g., grasp, insert.

# TELEROBOTIC SYSTEM TECHNOLOGY

## Shared Control Capability

Shared control: utilization of both autonomously generated data and data from human

- Cartesian tool frame merging of autonomous and teleoperation inputs in real time
- Allows operator to modify autonomous motion by using hand controllers
- Can partition task into autonomous and operator controlled parts, e.g., operator controlling motion along pin insertion axis and autonomous system controlling interaction forces and torques.

# TELEROBOTIC SYSTEM TECHNOLOGY

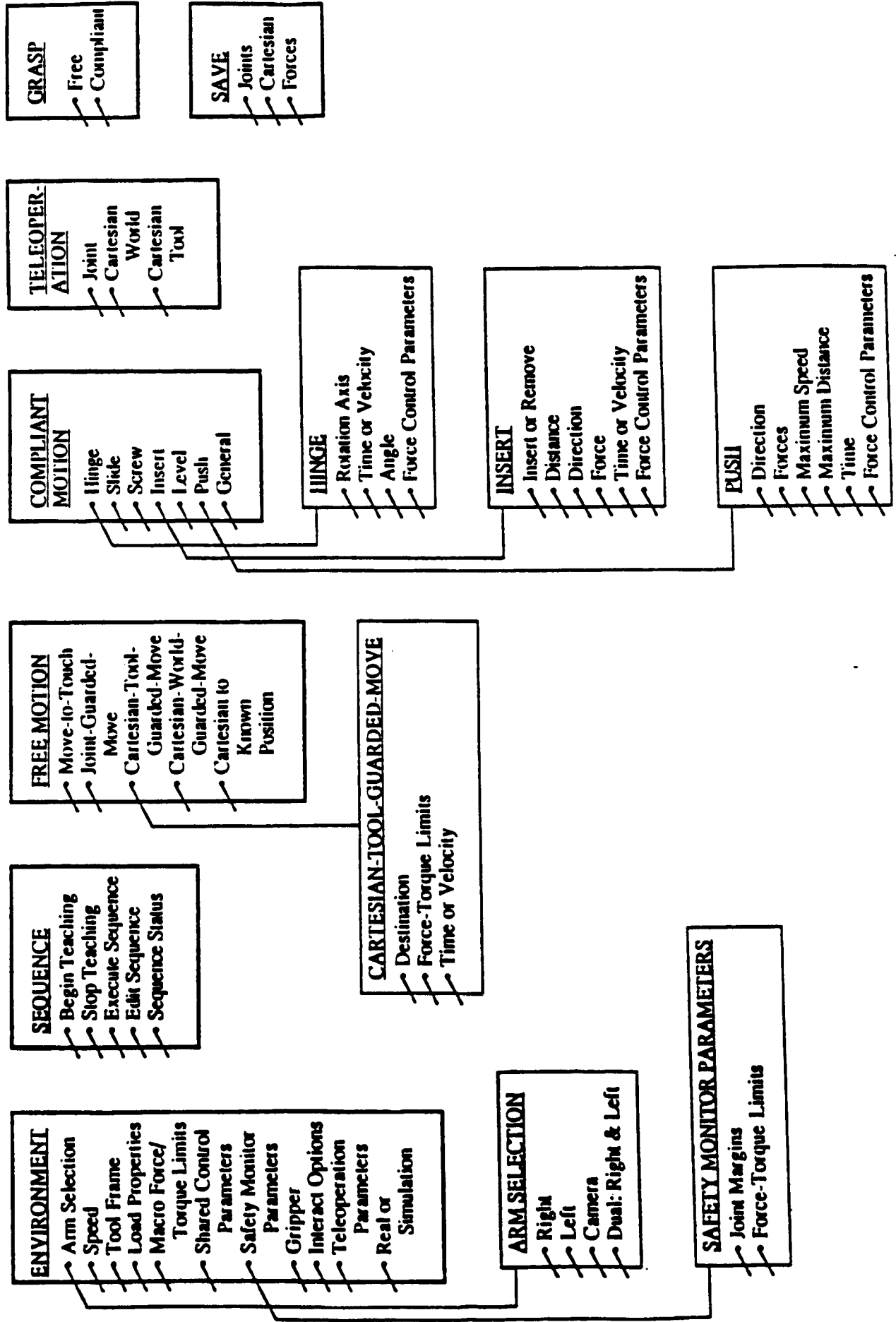
## Supervisory Control Capability

**Supervisory control:** Operator performs task planning, path planning, task specification, sequence building, error recovery, and replanning. **Computer** performs primitive task execution, force control, monitoring, and shared control.

- Developed operator menu interface to describe, execute, and monitor task execution.
- Operator selects autonomous, teleoperation, or shared control and related parameters.
- Autonomous system aids operator in parameter selection
- 3-D graphics simulator available to preview task execution before running on real arms
- Sequencing: operator can save one or more commands as a sequence for later execution

# TELEROBOTIC SYSTEM TECHNOLOGY

## Interactive User Interface Capability



# **TELEROBOTIC SYSTEM TECHNOLOGY**

## **Summary of Accomplishments**

- Developed a unified autonomous and teleoperated robot control system
- Designed and implemented traded and shared control
- Developed dual-arm force reflecting teleoperation
- Developed autonomous motion primitives
- Developed interactive supervisory task synthesis and execution
- Demonstrated above capabilities on realistic space-like hardware

# TELEROBOTIC SYSTEM TECHNOLOGY

## FY 90 Plans

- Time-delay capability: Fixed, variable, random
- Local-remote communication link
- Merging of teleoperation features into shared control
- Further develop autonomous control primitives for shared control task execution
- Enhance operator's menu interface
- Decompose specific space telerobot tasks into teleoperation, autonomous, and shared control subtasks and demonstrate system capability



# JPL/KSC TELEROBOTIC INSPECTION AND MANIPULATION DEMONSTRATION

February 8, 1990

JPL activity supported by Code ST

KSC activity supported by Code M



## TASK PARTICIPANTS

### JPL:

Wayne Schober, Program Manager  
Bert Hansen III, Technical Manager  
Brian Wilcox, Task Manager  
Bruce Bon, S&P Lead  
Jack Morrison, S&P S/W Engineer  
Todd Litwin, S&P S/W Engineer  
Steve Peters, TPR Lead  
David Mittman, TPR S/W Engineer  
Jacquie O'Meara, TPR S/W Engineer  
Carol Collins, TPR S/W Engineer

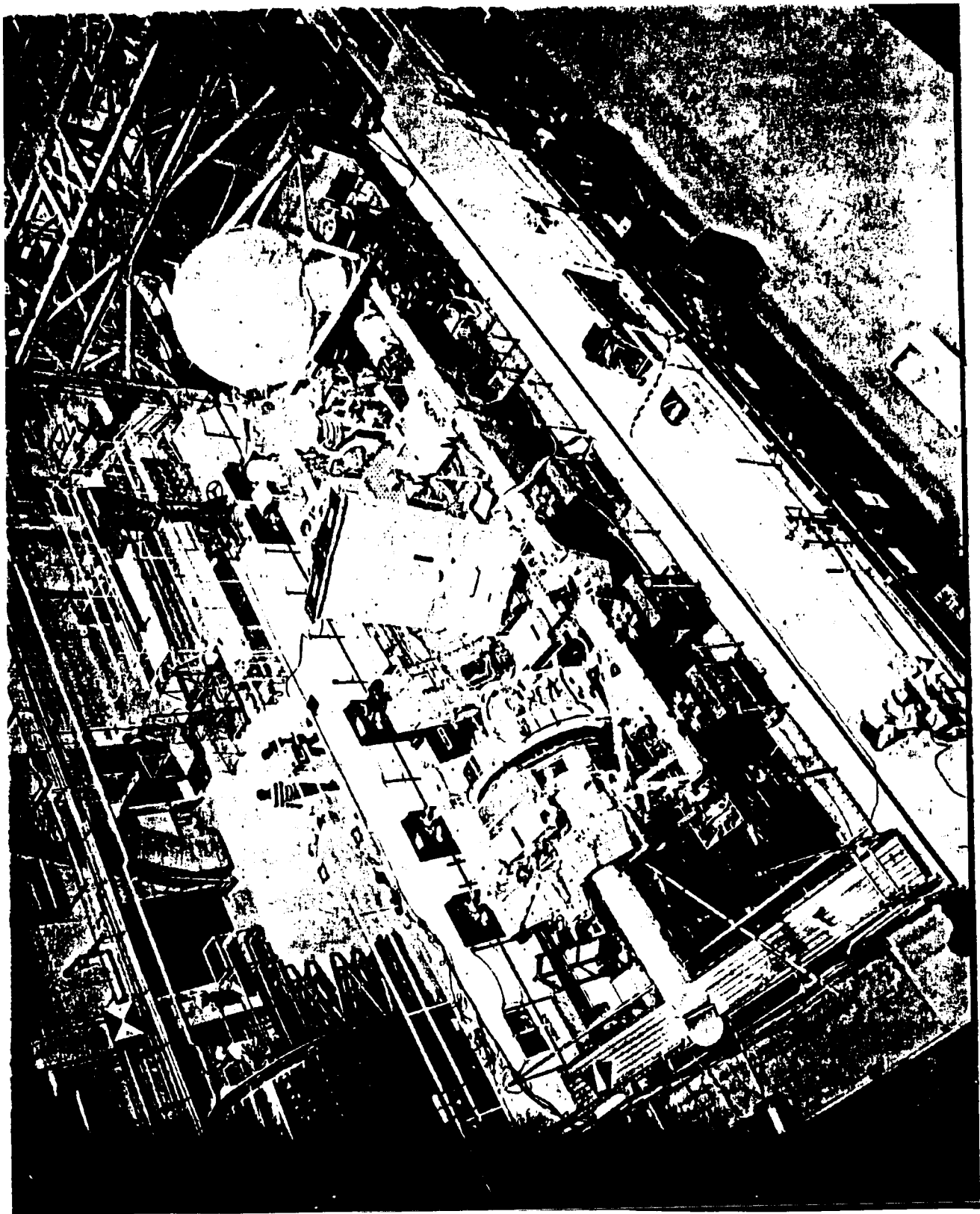
### KSC:

V. Leon Davis, Robotics Supervisor (NASA)  
Bob Lewis, Task Element Manager (NASA)  
Mike Sklar, Robotics Engineer (McDonnell Douglas)  
Dan Wegerif, Robotics Engineer (McDonnell Douglas)  
Alex Ladd, Lead S/W Engineer (Boeing)  
Jose Lago, ADC S/W Engineer (Boeing)  
James Spencer, Vision System S/W Engineer (Boeing)  
John Brogdon, S/W Engineer (Boeing)  
Bob Humeniuk, IGES CAD Modeler (McDonnell Douglas)



## CODE M PROBLEM DOMAIN -- PAYLOAD OPERATIONS

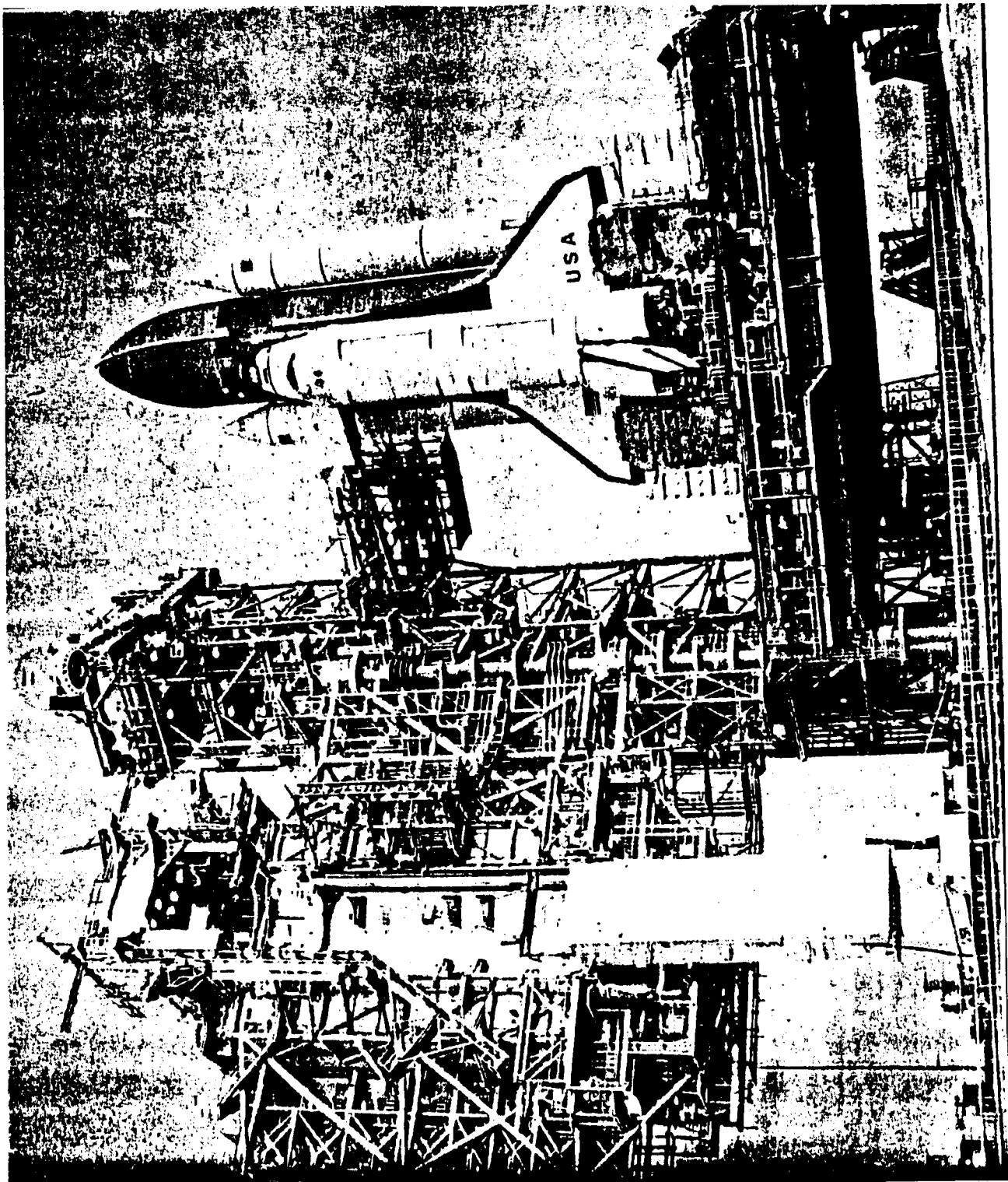
At KSC, access to payloads is restricted. At the Operation and Checkout Facilities, cages are sometimes built to lower a man down between satellites to retrieve, replace or connect an object. In this environment, there is room for access above the workstands.





## VERTICAL PAYLOADS AT THE LAUNCH PAD

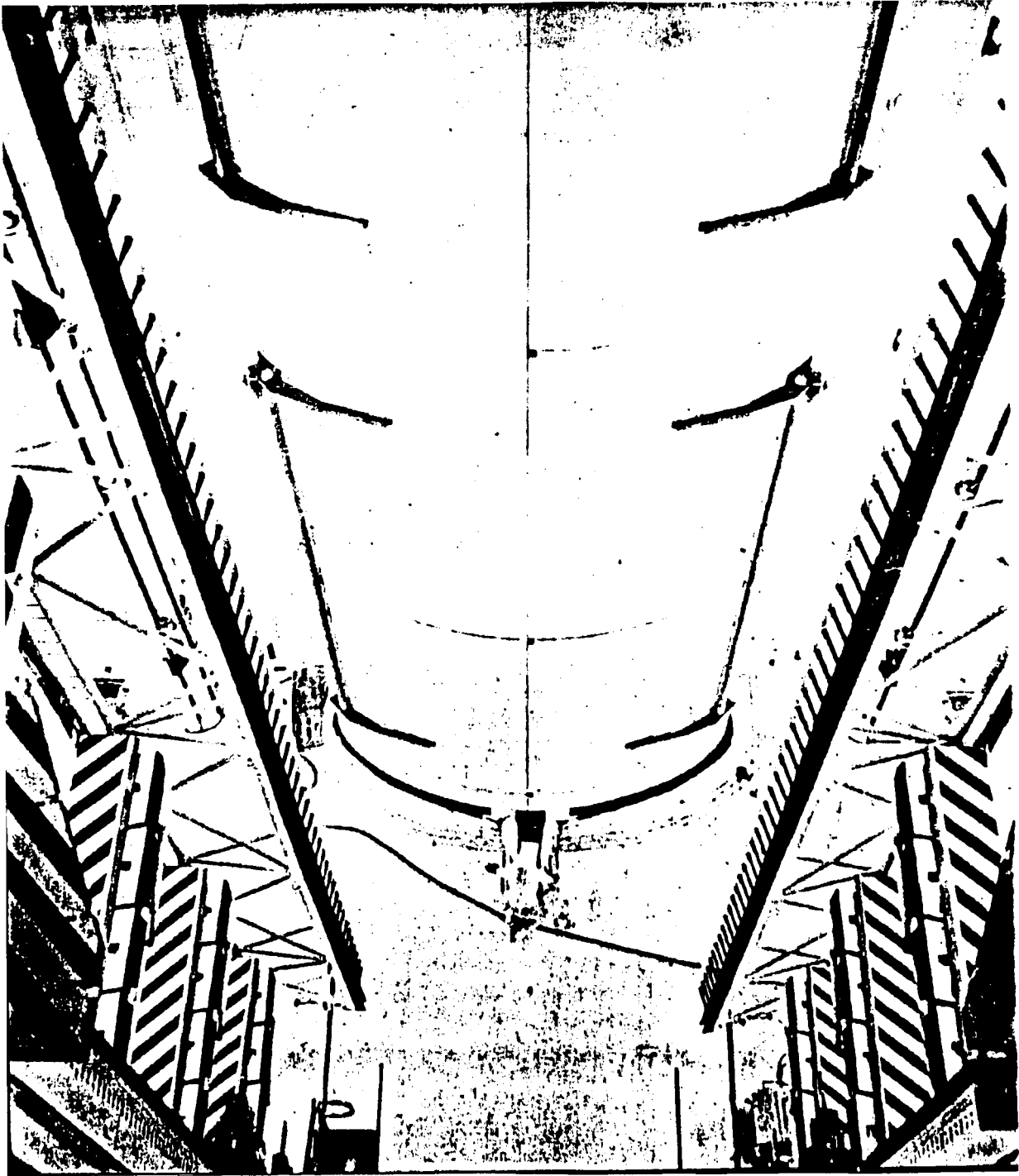
At the launch pad, payloads are lifted vertically inside a canister into the Payload Changeout Room (PCR).





## INSIDE THE PAYLOAD CHANGEOUT ROOM

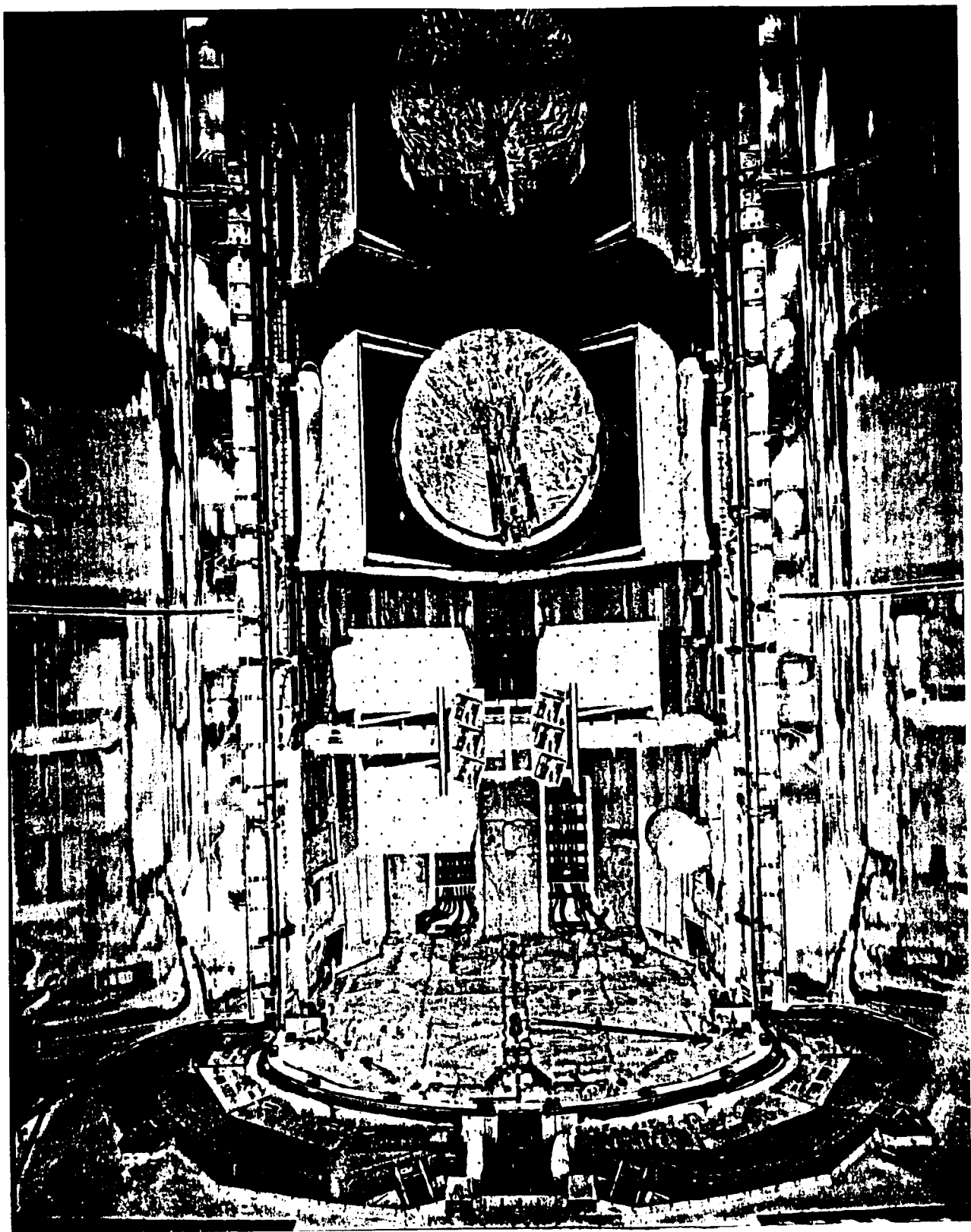
Inside the PCR, it seems like a lot of room is available.





## GETTING THE PAYLOAD INTO THE PCR

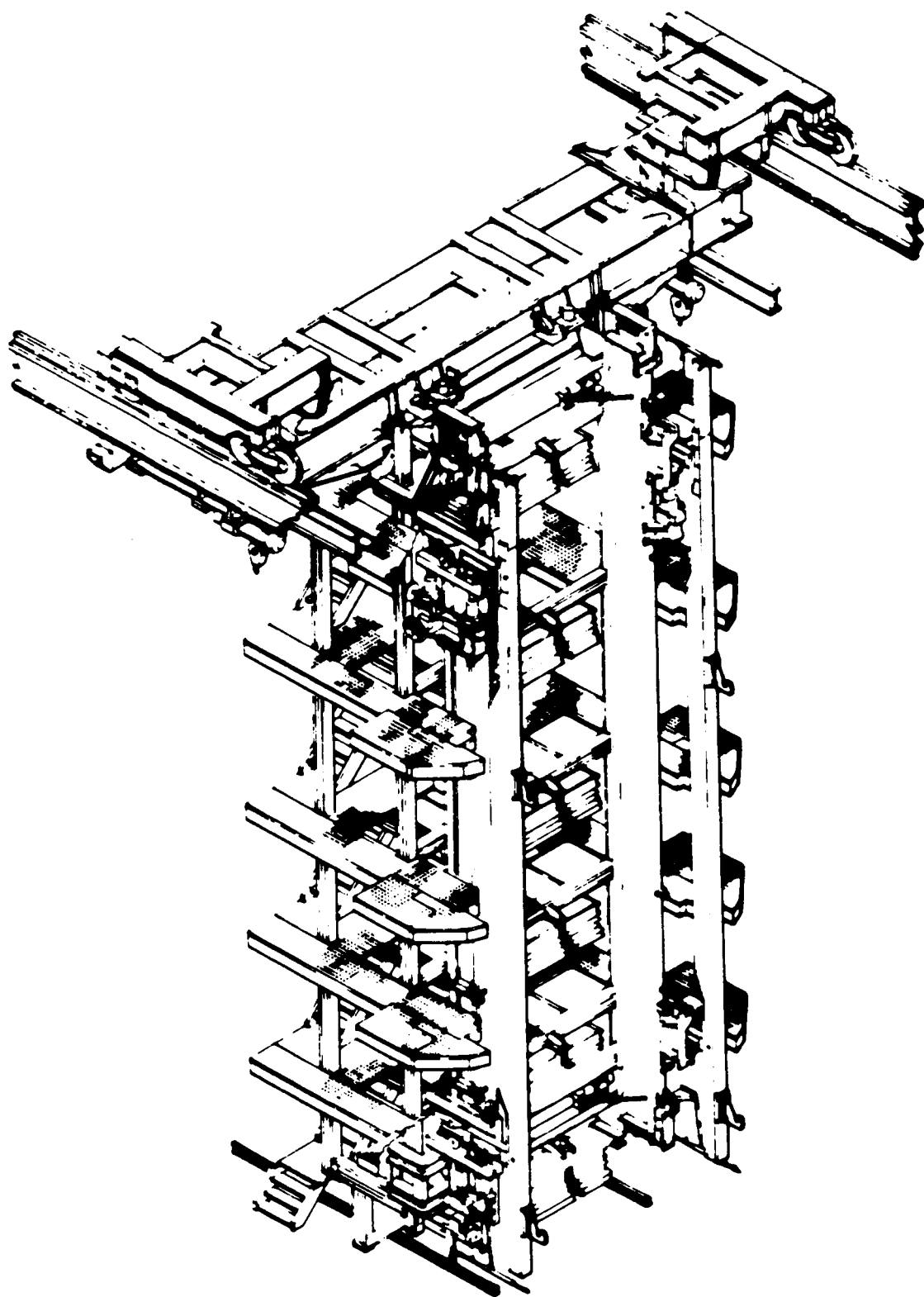
Before a payload is installed in the orbiter bay, it must be removed from the canister and brought inside the PCR. After the PCR and Rotating Service Structure is moved over/around the orbiter payload bay, the payload will be inserted into the orbiter bay. The payload shown here includes two satellites to be launched from the bay using PAM-D rocket motors.





## THE PAYLOAD GROUND HANDLING MECHANISM

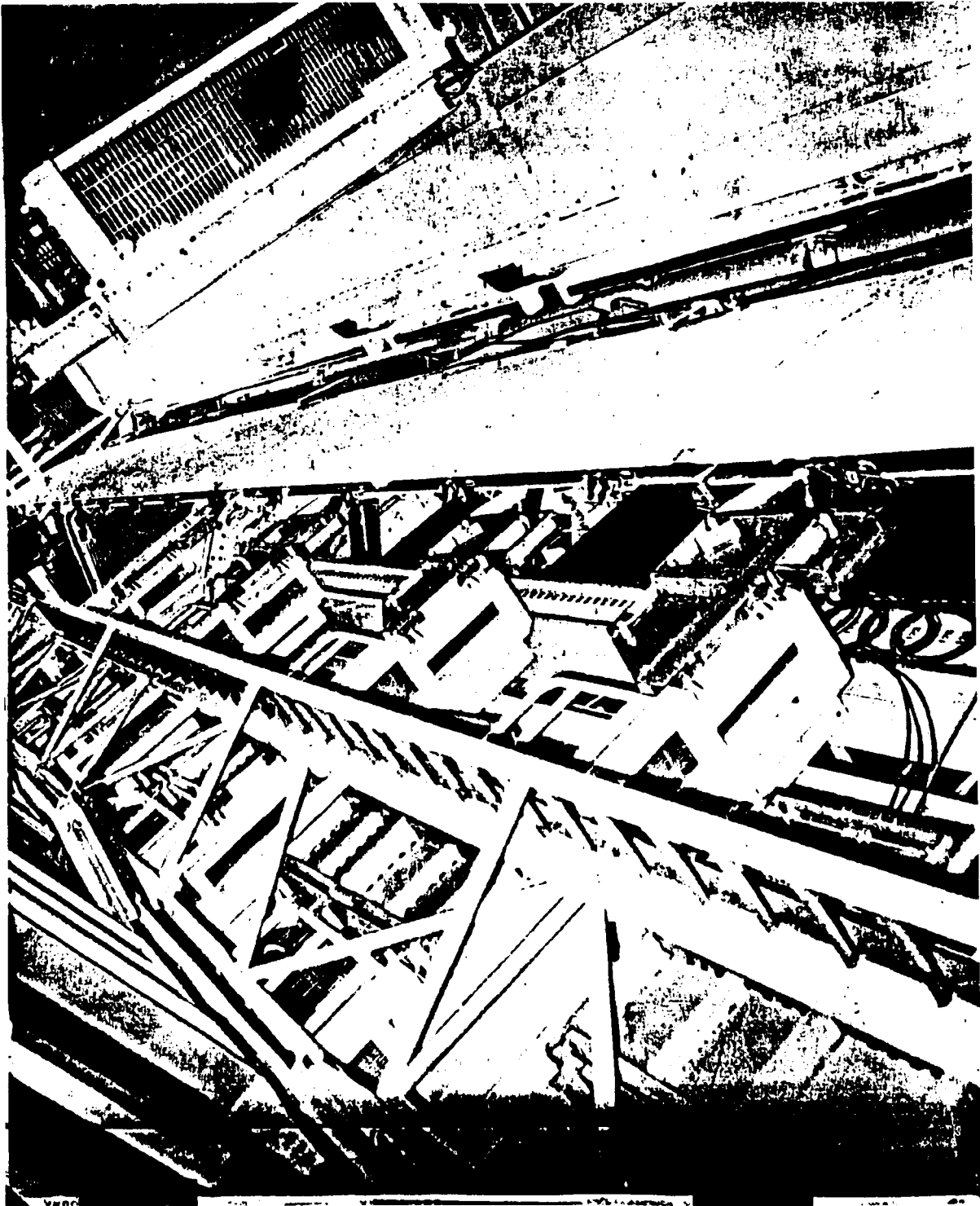
The Payload Ground Handling Mechanism (PGHM) is a floating device on an overhead beam that removes the payload from the canister and inserts it into the orbiter bay.





## PGHM OBSTRUCTION OF THE PAYLOAD

When the payload has been inserted into the orbiter bay, you cannot even see it behind the PGHM structure.





## PAYLOAD ACCESS IN THE PCR

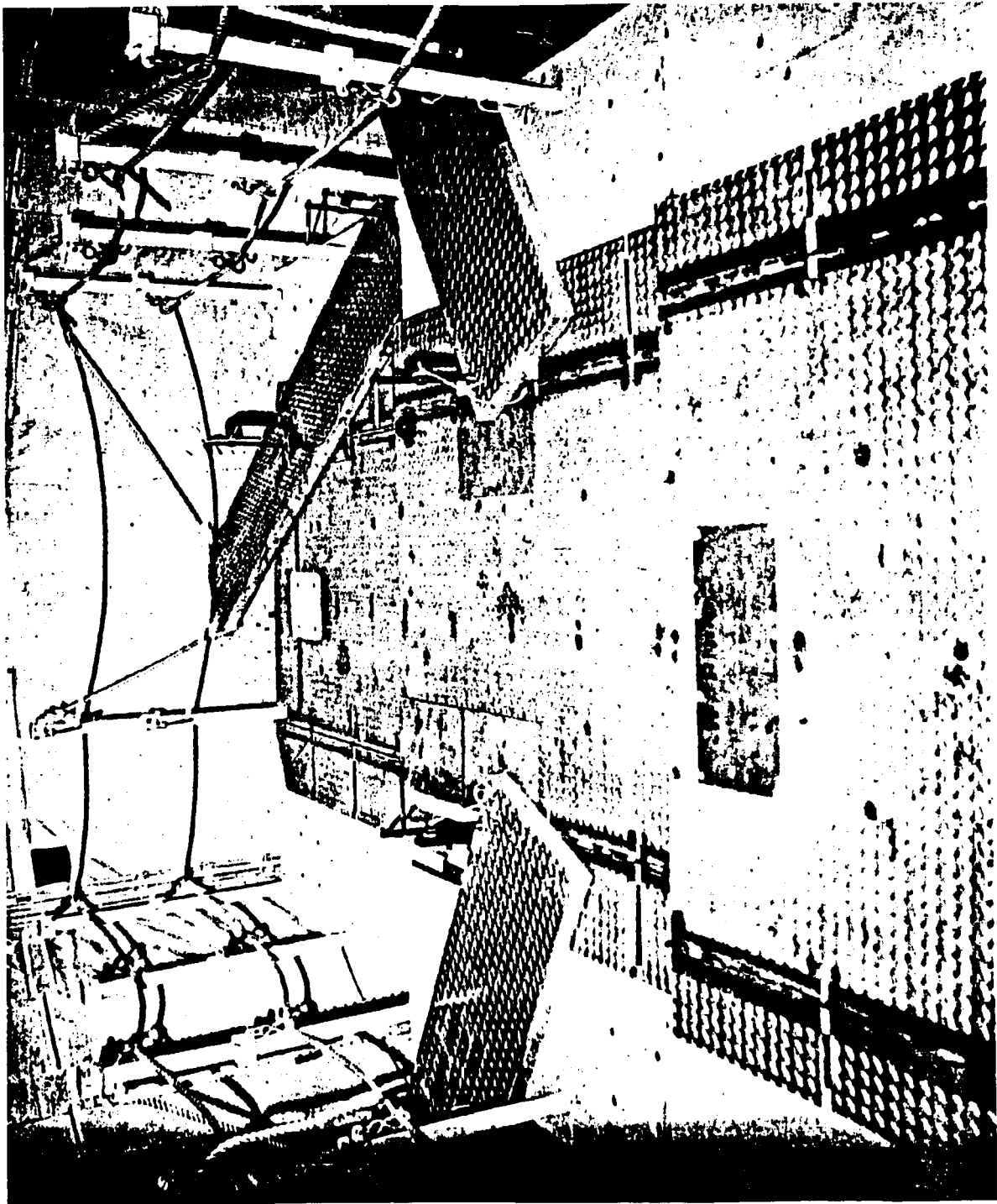
Limited access to the payload is possible by crawling out onto platforms.





## PAYLOAD ACCESS IN THE PCR

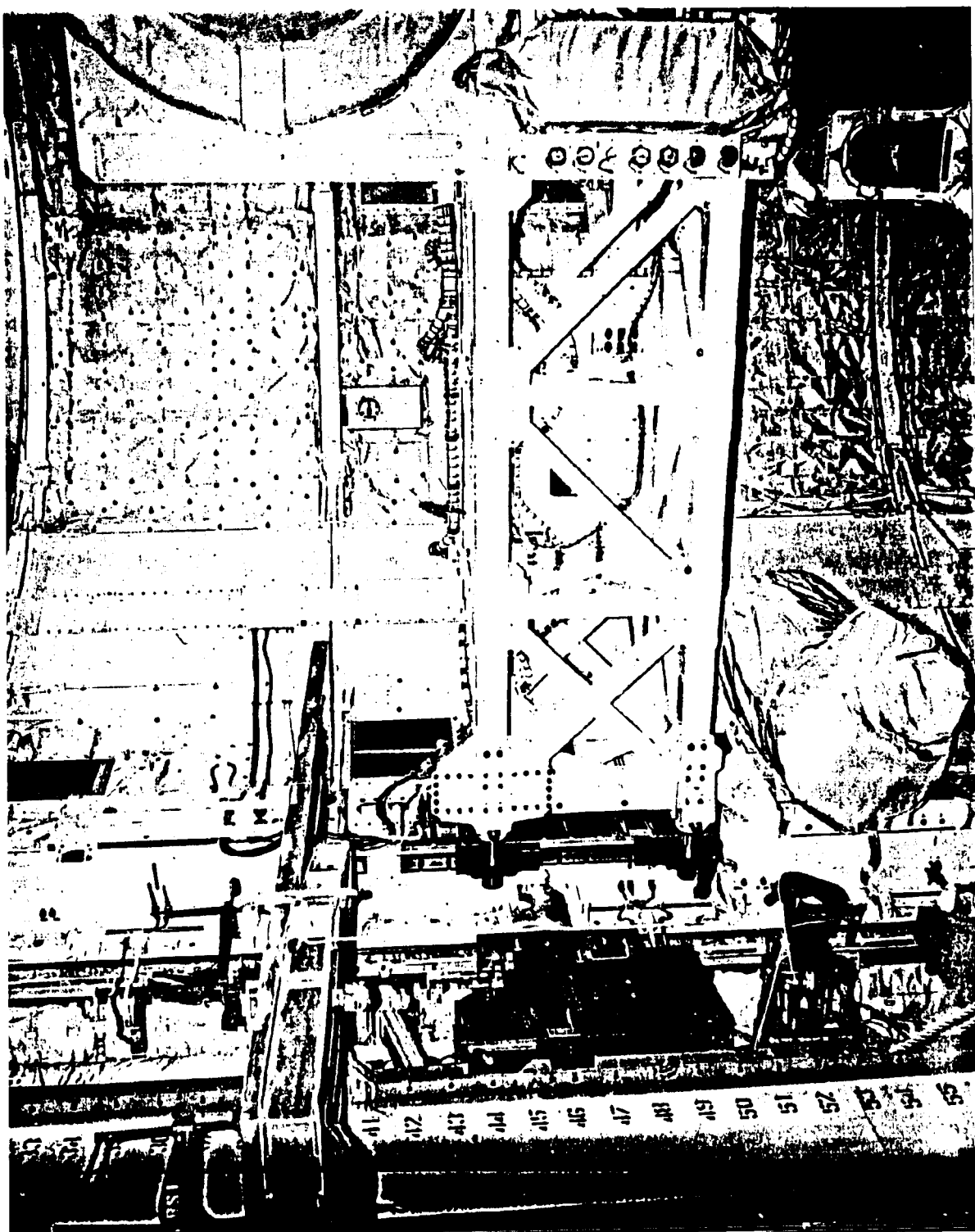
"C" clamps, gang planks and roll-out platforms are used to get access inside the orbiter bay.





## PAYLOAD ACCESS IN THE PCR

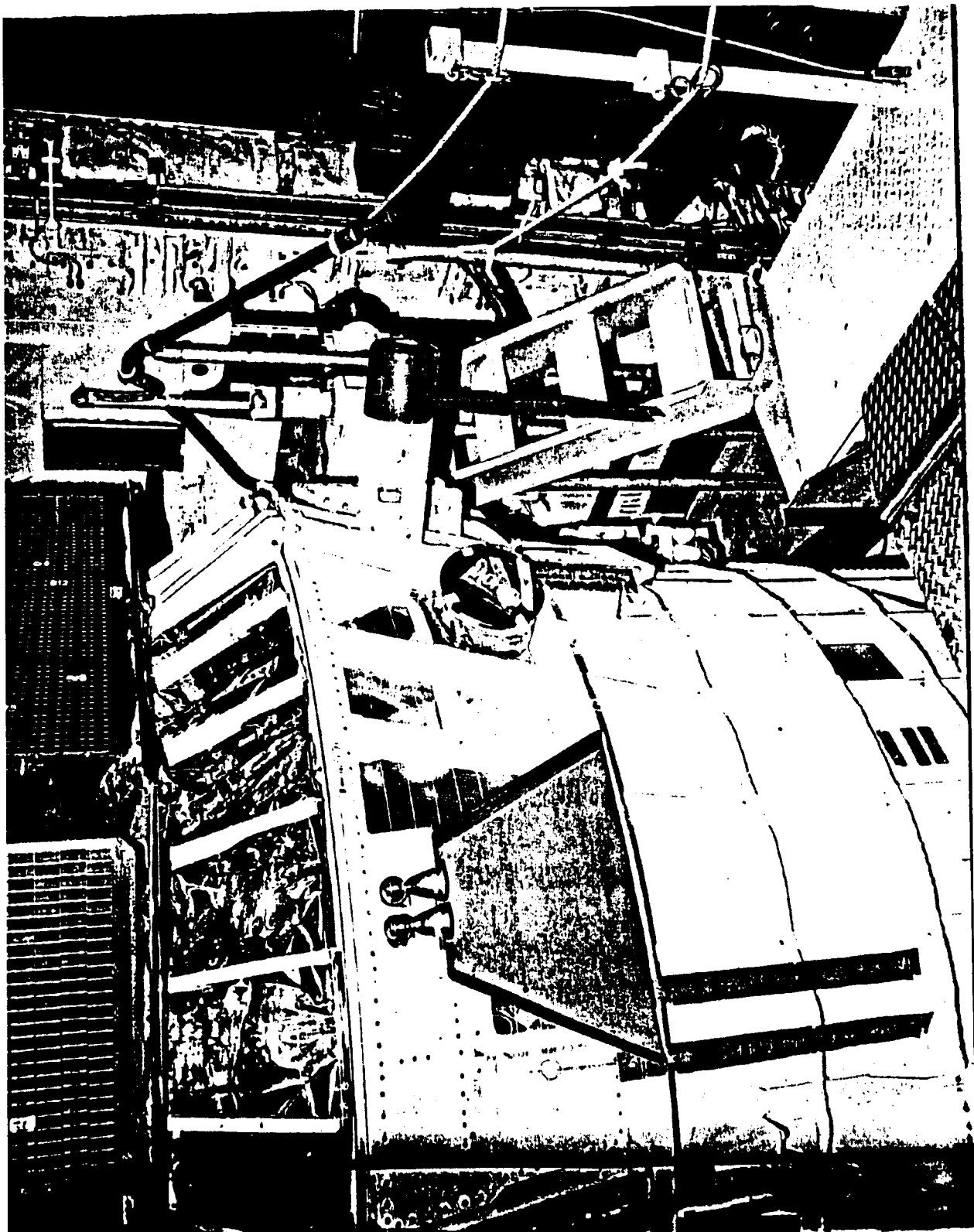
Sometimes a technician has to climb out onto a gang plank in order to take close-up photos or to remove lens dust covers . . .





## PAYLOAD ACCESS IN THE PCR

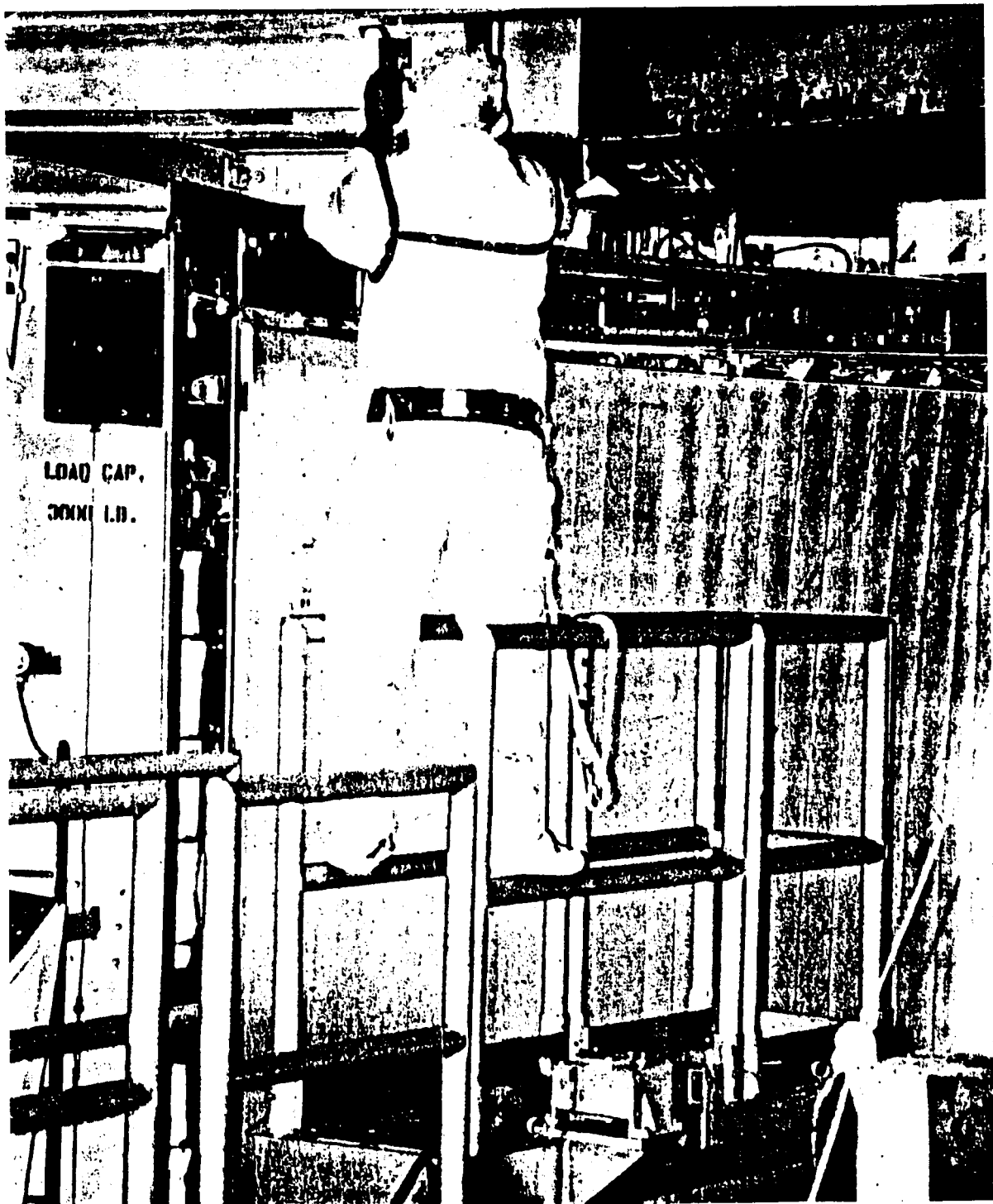
... or to remove tagged items just prior to launch.





## PAYLOAD ACCESS IN THE PCR

Twice for each launch, hazardous positions (65 feet above a multimillion dollar payload) have to be reached to attach grounding straps. This involves bolts and test gear that are not tethered and which, if dropped, may damage an expensive payload or experiment, with large "return from pad" consequences.





## TECHNICAL APPROACH -- RADL

This is a schematic of the Robotics Applications Development Laboratory (RADL) at KSC. It includes a large ASEA robot arm on a track and various support computers for controlling the arm and processing video data for machine vision applications.

# ROBOTICS APPLICATIONS DEVELOPMENT LABORATORY (RADL)

EAR 1 DEMO:

PAYLOAD BAY  
MOCKUP

LARGE  
ASEA ROBOT

KSC

LEASED LINE TO KSC MICRO VAX  
AND DATACUBE VISION CONTROL  
SYSTEM

9600 BAUD  
LINE

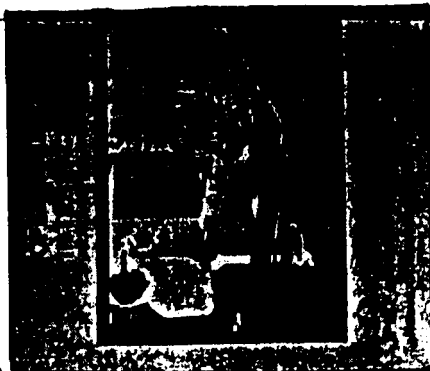
JPL

JPL

KSC

MOCKUP

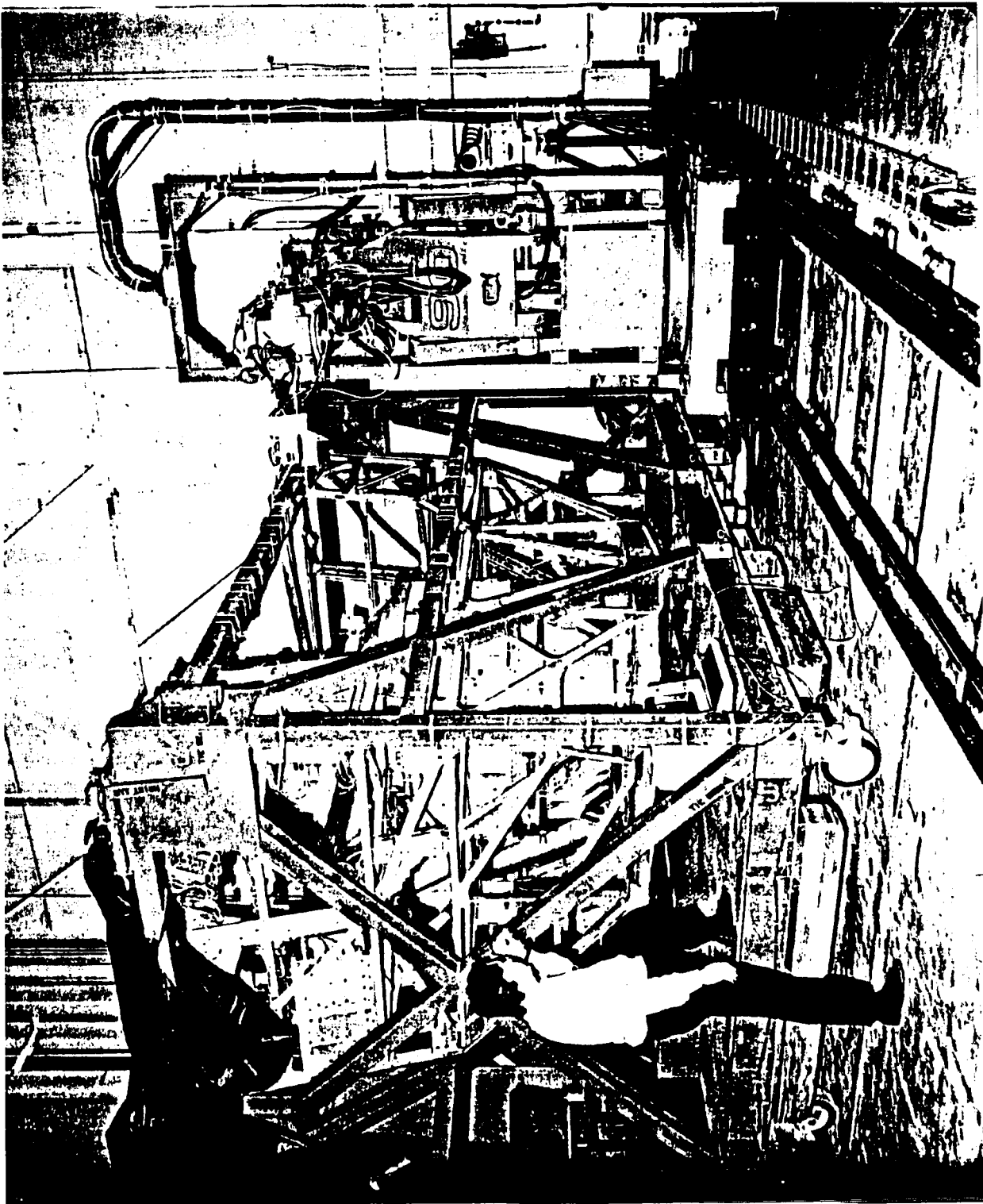
CONTROL





## TECHNICAL APPROACH -- WORKSITE

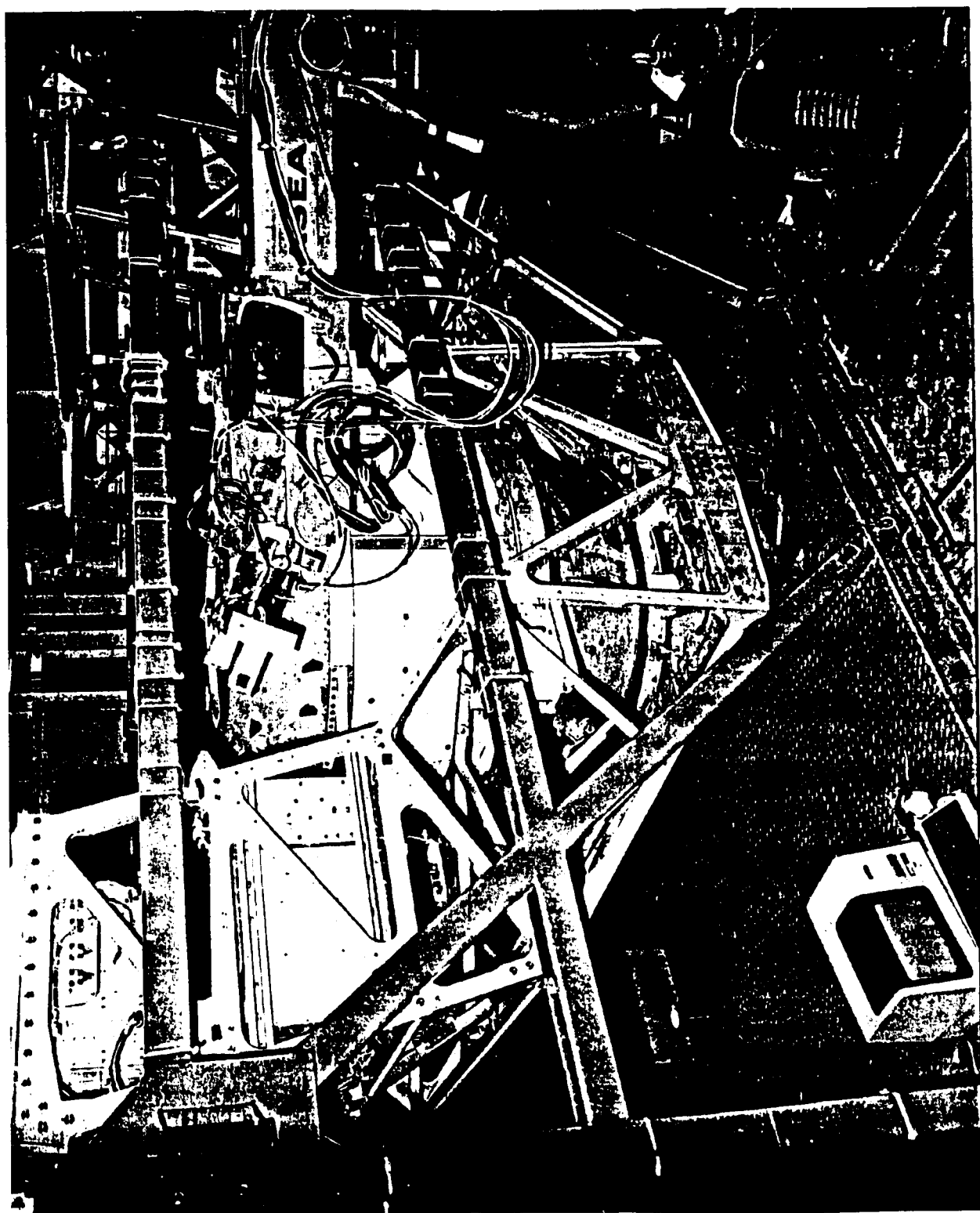
This shows an overview of the worksite, with the inert PAM-D motor and frame, and the ASEA robot arm.





## TECHNICAL APPROACH -- WORKSITE

This is a closer view of the worksite, showing the dual-camera platform and the gap in the frame through which the arm must move to accomplish the task. Note the small clearances relative to the size of the arm.





## CODE M OBJECTIVES

Demonstrate effective man/machine teamwork for payload processing:

- Effectiveness of telerobotic operations on real tasks
- Safety of telerobotic operations for personnel & payload
- Productivity of telerobotic operations
- Reliability of telerobotic operations



## CODE ST PROBLEM DOMAIN: SPACE STATION EXTERNAL OPERATIONS ALTERNATIVES

- Extravehicular Activity (EVA) -- astronauts in spacesuits
- Flight Telerobot Servicer (FTS) -- teleoperation with astronauts controlling from Space Station
- Ground/Remote Telerobotics -- operators on ground controlling semi-autonomous telerobot at the Space Station

Real operations will include a mixture of these operations modes



## **CODE ST PROBLEM DOMAIN: EXTRAVEHICULAR ACTIVITY**

### **ADVANTAGE:**

- Astronaut at the work-site -- perception, problem-solving

### **DISADVANTAGES:**

- Safety issue -- astronaut risk is inherent in EVA
- Low productivity due to limitations of space suit
- Requires large amounts of expensive astronaut on-orbit time
- Mistakes due to limited dexterity reduce reliability and safety



## CODE ST PROBLEM DOMAIN: FTS/TELEOPERATION

### ADVANTAGE:

- No EVA astronaut risk

### DISADVANTAGES:

- Low productivity due to limitations of teleoperation
- Requires large amounts of expensive astronaut on-orbit time
- Mistakes due to limited dexterity reduce reliability and safety



## CODE ST PROBLEM DOMAIN: GROUND/REMOTE TELEROBOTICS

### ADVANTAGES:

- No EVA astronaut risk
- Enhanced productivity due to more available work time per day and partial autonomy
- Astronaut on-orbit time not required -- control from the ground
- Enhanced reliability

### DISADVANTAGE:

- Operator has limited perception of problems because of remoteness from task

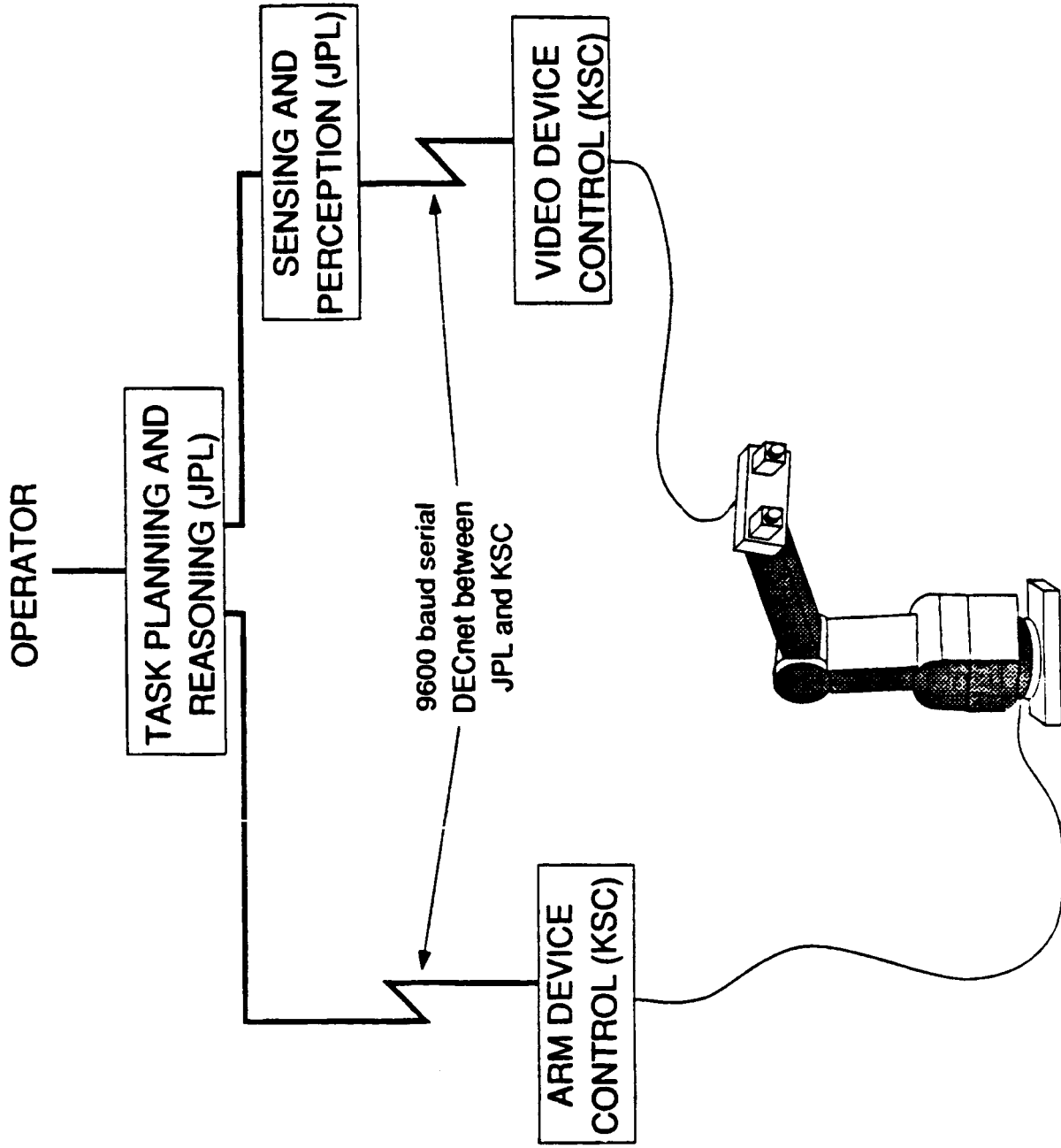


## CODE ST OBJECTIVES

Demonstrate effective telerobotic space operations in the presence of communications delays and low bandwidth:

- Effectiveness of telerobotic operations in the presence of substantial and variable communications delays
- Effectiveness of telerobotic operations on real tasks
- Safety of telerobotic operations for personnel & payload
- Productivity of telerobotic operations
- Reliability of telerobotic operations

# LOGICAL BLOCK DIAGRAM





## TECHNICAL APPROACH -- SUMMARY

- Robot arm, video cameras & PAM-D frame at KSC
- Operator interface, spatial planning & machine vision at JPL
- Slow network link provides communications between JPL and KSC
- Operator interface, spatial planning & machine vision technology transferred from Code R Telerobot
- New work involved new interfaces, video processing, device control software at KSC, calibration and modeling, and spatial planning



## SUMMARY OF FIRST-YEAR RESULTS

- Hardware and communications installed and integrated
- PAM-D acquired and IGES models generated
- Device control software developed at KSC
- JPL Sensing & Perception (S&P) Subsystem modified
- JPL Task Planning and Reasoning (TPR) Subsystem modified and extended for spatial planning, direct interface to KSC, and graphic user interface
- Successful capability test, including planned robot arm motion into an occluded region of real flight hardware, with machine vision verification



## DIFFICULTIES OVERCOME

- Network communications difficulties
- ASEA robot arm controller limitations
- IGES model conversion and transformation to robot coordinate frame
- Camera calibration with poor dispersion of calibration points



## BASELINE IMPACTS ON SPACE STATION INITIAL OPERATIONAL CAPABILITY (IOC)

Possible modifications in requirements:

- Communications with Earth
- Modifications to FTS
- Operator control facilities on Earth
- FTS tools and jigs

Benefits:

- Continuous station assembly controlled from the ground
- Improved astronaut safety through reduced EVA
- Improved reliability on repetitive, tiring tasks



## JPL FUTURE PLANS: FY 1990

- Expanded world model
- Improved system speed and operator interface
- Fine-motion planning to enable arbitrary motions
- Develop models of modified arm and cameras
- Begin transition to VME-based hardware



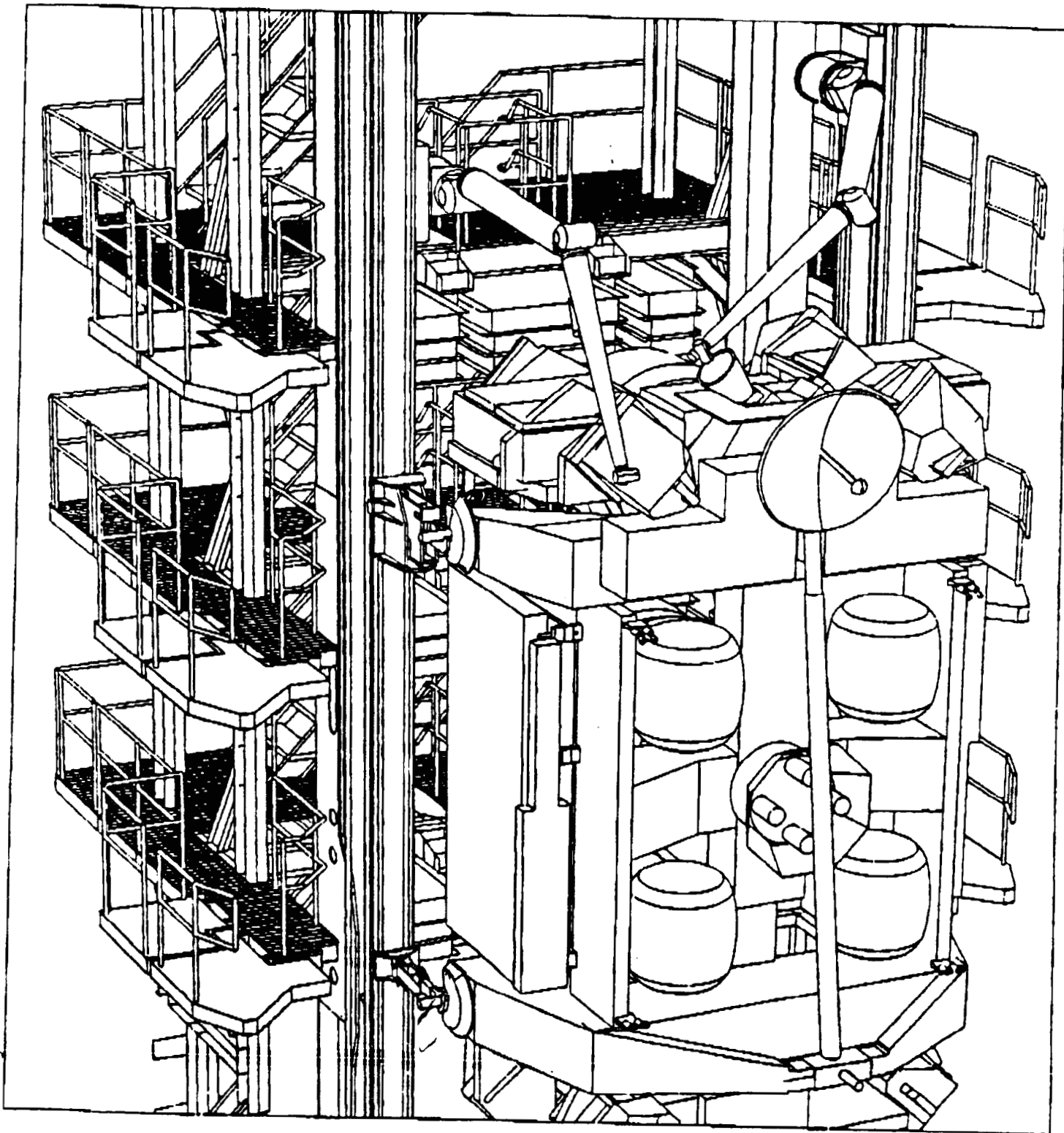
## KSC FUTURE PLANS: FY 1990

- Design proximity sensors for robot arm
- Develop articulated extension to ASEA arm
- Develop camera system for modified arm
- Develop kinematic calibration of ASEA arm
- Install TPR software at KSC

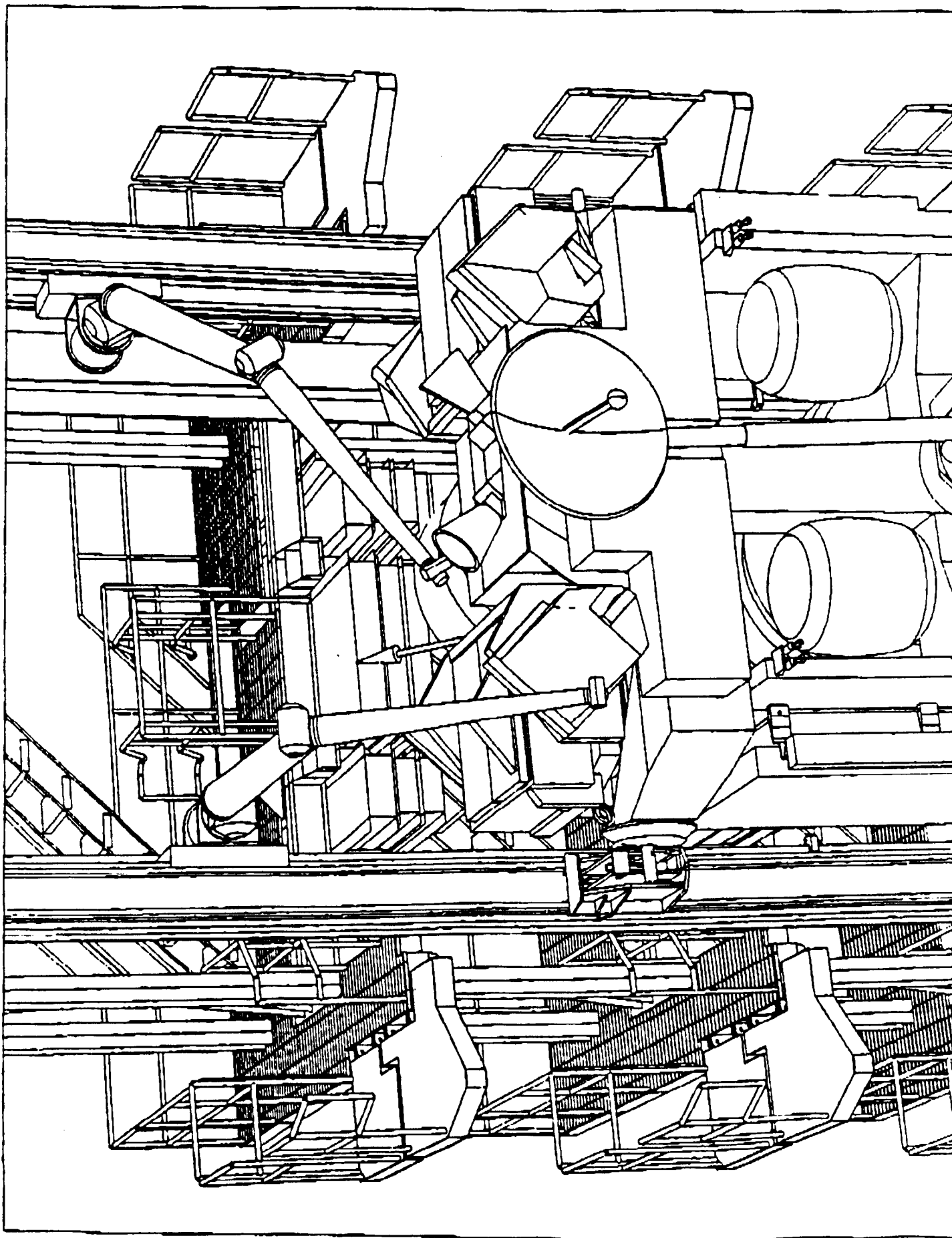


## GROUND PROCESSING APPLICATIONS PAYLOAD CHANGEOUT ROOM PROCESSING ROBOT

- Current technology directly applicable to shuttle launch pad Payload Processing Robot
- Conceptual system consists of:
  - 2 mobile base arms mounted on Payload Ground Handling Mechanism at the Payload Changeout Room
  - High-level user interface allows operation by current technicians
  - Intelligent path planning based on CAD models of facility, orbiter and payloads
  - All facilities and a majority of payloads are currently modeled
  - Redundant obstacle avoidance, critical for payload operations, provided by vision system and proximity sensing system



**CONCEPTUAL PCR PROCESSING ROBOT**



GAMMA RAY OBSERVATORY PAD PROCESSING



## PCR ROBOT BENEFITS

- Reduced dependence/cost for special access platforms and clamped diving boards
- Greater degree of access than currently available
- Potential reduced process flow time



## FUTURE CHALLENGES:

- Proximity sensing
- Spatial planning
- Perception
- Dexterous manipulation

# **ROBOTIC ASSEMBLY OF LARGE SPACE STRUCTURES**

**RALPH W. WILL AND MARVIN D. RHODES**

**NASA Langley Research Center  
Hampton, VA 23665**

## **ABSTRACT**

The Automated Structures Assembly Laboratory (ASAL) has been developed at Langley Research Center for the purpose of identifying the problems associated with assembling large space structures using robotic manipulators, investigating systems and techniques applicable to such assembly tasks, and developing methodology for an in-space construction facility. The ASAL facility and its robotic manipulator system is described and a discussion of the initial series of assembly operations are included. The status of the system software development is summarized and an outline for follow-on testing and expanded capability is given.

## **ROBOTIC ASSEMBLY TESTBED**

### **Objectives:**

- Identification of problems associated with assembly large space structures using robotic manipulators.
- Evaluation of systems and techniques applicable to space assembly tasks.
- Development of methodology for an in-space construction facility.

### **Approach:**

- Utilize standard industrial robot mounted on moving base to assemble simple tetrahedral truss.
- Evaluate sensor hardware, assembly techniques and planning systems.

## **ROBOTIC ASSEMBLY TESTBED**

### **Objectives and Approach:**

Most work in the robotic in-space assembly area has been confined to the study stage. Langley Research Center has developed a robotic assembly testbed to investigate automated assembly through actual hardware experience. Such experiments should identify real problems associated with the robotic assembly of space structures. The testbed facility utilizes a standard industrial manipulator mounted on a precisely-controlled motion base to assemble a simple tetrahedral truss made up of 102 two-meter-long struts. The program involves the progressive incorporation of robotics technology in areas such as sensors and planners to evaluate both systems and technique for automated assembly.

The Automated Structures assembly program, a joint effort involving both the Structures and Flight Systems Directorates at LaRC, is also aimed at developing a generalized robotic assembly capability by considering other structural configurations and other assembly tasks such as module, utility and panel installation. This progressive approach to systems development and task capability is expected to generate generalized robotic assembly methodology for an in-space construction facility.

## **ROBOTIC ASSEMBLY TESTBED**

### **Accomplishments:**

- Testbed facility operational with motion base accuracies of .002".
- Development of truss assembly sequence and naming convention.
- Robotic assembly of 24-member truss ring using force/torque feedback.
- Preliminary evaluation of position and ranging sensor guidance.
- Development of robotic end effector for more general assembly tasks.

## **ROBOTIC ASSEMBLY TESTBED**

### **Accomplishments:**

The robotic assembly testbed facility has been operational since January 1989 with a motion base positioning accuracy significantly exceeding specifications. Assembling testing began in February using manually determined assembly sequences for the tetrahedral truss structure. The 24-member inner ring of the structure was successfully assembled after several modifications to the assembly sequence. A force/torque feedback algorithm was also developed for these tests when strut-node joining operations proved unreliable without sensor feedback. The sensor work has been continued in preliminary evaluations of position and ranging sensor hardware and guidance algorithms involving these sensors.

A more generalized robotic end effector has been designed and should be operational by the end of the year. This end effector handles the struts one end at a time and is thus capable of installing struts of varying lengths, as well as suitably equipped modules and panels. This end effector also incorporates all improvements identified during the initial assembly tests. Robot positioning, collision-free paths, and assembly sequences have been determined for the 78-strut second ring of the tetrahedral truss and assembly testing is underway. For these tests, fully automated software with error recovery capability is being used.

## **ROBOTIC ASSEMBLY TESTBED**

### **Assembly Test Observations:**

- Testbed facility motion base and manipulator accuracy is adequate.
- Truss/node joining concept of inserting strut into a captured node is necessary.
- Full end effector instrumentation and TV monitoring is essential.
- Force/torque feedback to achieve .001" positioning accuracy is necessary for joining.
- Positive actuation, as opposed to passive springs, is required for reliable end effector operation.
- Computer control is essential - details of assembly exceed operator ability.

## **ROBOTIC ASSEMBLY TESTBED**

### **Assembly Test Observations:**

The initial 24-meter truss assembly tests yielded several significant results, some expected and some not. The testbed facility itself with its .002" positioning accuracy proved to be more than adequate for these operations. The strut-node joining operation, however, required considerably more positioning accuracy than anticipated. A force-torque feedback system was developed, providing an accuracy of .001", a level required for stable, precise structures. These accuracy requirements emphasize the necessity for a truss-node joining concept which involves inserting a strut into a captured node. This joining hardware must utilize positive actuation rather than passive spring-type mechanisms in order to insure reliable operation with the accuracy constraints.

It also became apparent that full instrumentation of all end effector operations is essential for automated assembly. End effector-mounted TV cameras are also needed for operator monitoring and verification of end effector functions. Signal line limitations for end effector instrumentation and position sensors make on-board processing for assembly end effectors very attractive. The complexity of the assembly operations and the amount of information involved exceeds the capability of human operators. The great majority of failures encountered during the initial assembly tests were directly attributable to operator error.

## **ROBOTIC ASSEMBLY TESTBED**

### **Future Research Activities:**

- Develop sensor guidance techniques for assembly operations.
- Evaluate generalized end effector for a variety of tasks including module installation and curved truss assembly.
- Develop path planning and sequence planning techniques.
- Evaluate 3-D graphics interface for operator monitoring and automated planning.
- Investigate coordinated motions of manipulator and motion base.

## **ROBOTIC ASSEMBLY TESTBED**

### **Future Research Activities:**

After completion of the complete 102-member tetrahedral truss assembly, the research on assembly systems and techniques will continue with the incorporation of the generalized single-ended end effector into the assembly system. At the same time, positioning and ranging sensor feedback will be introduced. This will eliminate the need to teach strut installation points by using sensors to track the node positions. A three level guidance technique utilizing knowledge of the structure's geometry, position sensors and the current force/torque sensors will be evaluated for strut assembly. Path planning and collision avoidance algorithms based on structural geometry and supplemented by sensor information will be evaluated.

More advanced planning systems to generate assembly sequences and alternatives in the event of failures will be developed along with 3-D graphics display techniques for operator monitoring of the planning systems. Other structural configurations involving varying length struts and module or panel installation tasks will be evaluated in the automated planning environment. These more complex tasks may also involve dynamic coordination between the robot arm, the end effector, and the motion base.

**JPL**

---

**Telerobotics / EVA**

**Joint Analysis System**

**(TEJAS)**

**A Standardized Technology Analysis Tool  
for EVA and Telerobotics**

**Presentation to NASA Code ST**

**Michael L. Drews    Section 347    11/9/89**

**JPL**

---

## **JPL** What Is TEJAS?

---

### **TEJAS Is...**

- an information system of related multi-media data
- an task analysis tool for EVA and Telerobotics, that can form relational links between the two fields for any task, through their task primitives
- a technology development planning tool for Telerobotics, to give definitive direction for the needed technology push
- a disk-based reference for EVA and Telerobotics
- a set of hypermedia stacks that can be used individually, collectively, or card-by-card in many other applications

...built within an object-oriented programming environment that comes  
FREE with and runs on any Macintosh personal computer (≥2 Megs RAM)

## **JPL** TEJAS FY 89 Participants

---

Wayne Zimmerman: tool specification, basic system design, periodic review, project management

Dr. Bert Hansen: project management (JPL Code ST Projects)

Michael L. Drews: tool specification, software/hardware selection, basic system design, detailed stack designs & scripting, stack development, project management, reporting

Doug McAfee: TeleSystems stack detailed design and implementation

Paolo Fiorini: TeleTechnologies definitions and forecasting

Dr. Paul Schenker: TeleTechnologies definitions and forecasting

Jeff H. Smith: interactive definition and compilation of Telerobotics and EVA task primitives, beta-test/review of TelePrimitives and EVAPrimitives stacks

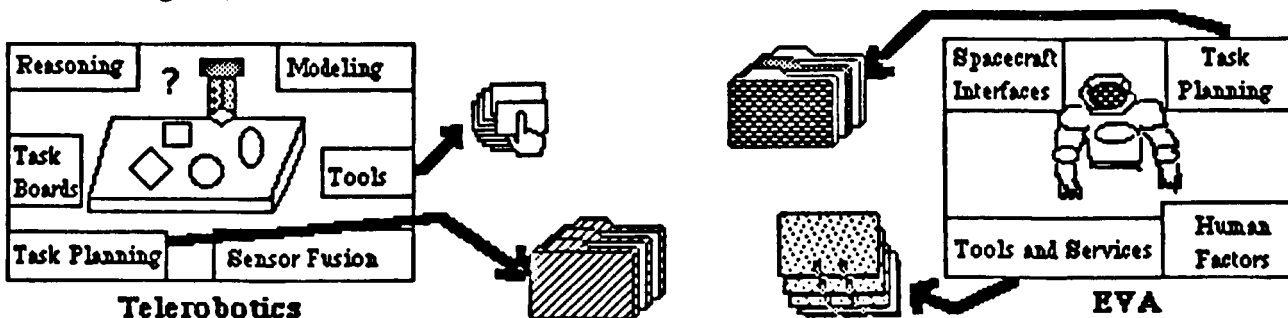
## **JPL** TEJAS Version 1.0 Features

- Relates EVA and Telerobotics through their specific task primitives. Clearly defines the generic primitive root, with a history and syntax (guide to usage).
- Interrelates EVA or Telerobotic verb primitives through use of Antonym and Like Verb fields.
- Interrelates the Description and Outline, Assumptions, and Open Issues for any task, EVA or Telerobotic, then aids in building a specific primitives list to approximate the Outline. Can link and compare any of these between tasks.
- Relates any telerobotic task and the associated assumptions to the Technologies required to perform that task under the stated circumstances.
- Provides detailed definition and forecasting on a 8-level readiness system for all telerobotics technologies in the database.

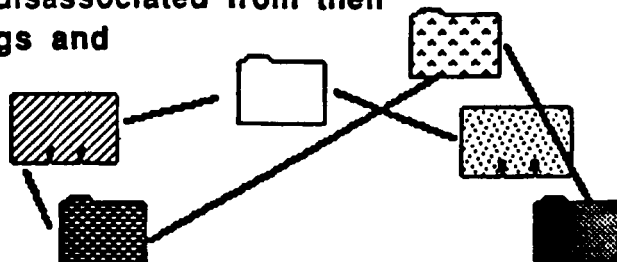
THE ENTIRE INFORMATION SYSTEM IS DYNAMICALLY UPDATEABLE:  
ADD AND ANALYZE ANY TECHNOLOGY, PRIMITIVE, ETC. IMMEDIATELY.

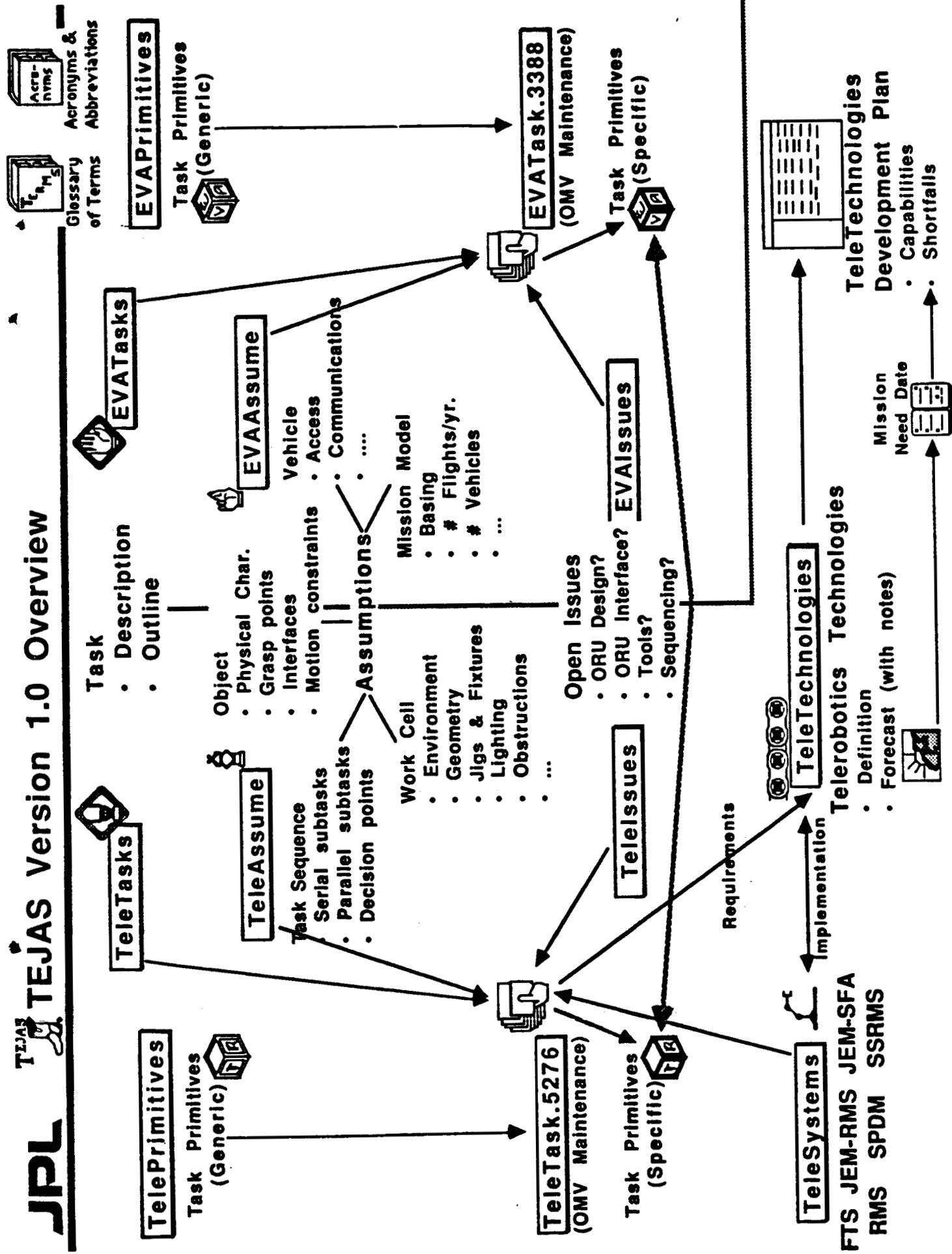
## **JPL** TEJAS Basic Approach

TEJAS takes elemental research components, such as task primitives and technologies, and treats them as cards in shuffled decks:



The cards can be disassociated from their traditional groupings and related freely, using the TEJAS scripts.





## **JPL** TEJAS Version 1.0 Reference Stacks Contents



**EVA Primitives (EVAPrimitives):** first-ever compilation of verb-oriented task primitives from major EVA Task Analysis Methodologies (TAMs) nationwide. Includes Verb, Definition, generic Syntax, Like Verbs (related terms, NOT synonyms), Antonym, and History fields. Helps standardize EVA TAM primitives, relate methodologies, and eliminate redundancies.

**Telerobotics Primitives (TelePrimitives):** same information fields as above, but from telerobotics TAMs. More primitives than for EVA, more exact.

**Telerobotics Technologies (TeleTechnologies):** individual definitions and forecasts (with histograms), via an 8-level technology readiness system, of all space telerobotics technologies. Includes NASREM area.

**Telerobotics and EVA Glossary of Terms (Glossary):** over 1100 commonly-used (and abused) terms, with up to three definitions (each with a source).

**Acronyms and Abbreviations Dictionary (Acronym):** over 800 entries, up to five definitions (each with a source), from NASA & Industry documents.

## **JPL** TEJAS Version 1.0 Analysis Stacks Contents



**Task Summaries (EYATasks, TeleTasks):** card summaries (one per task) of any EYA or Telerobotics task. Includes Task Name, Description, Outline.

**Task Assumptions (EYAAssume, TeleAssume):** cards for listing assumptions about workplace, task flow, tools, ORU's, etc. Aids in technology requirements analysis. One card per task.

**Task Open Issues (EYAIssues, TeleIssues):** cards for listing what is NOT resolved about a task being analyzed. One card per task.

**Specific Task stacks (e.g. EYATask.5267, TeleTask.5267):** created from Task Summaries, each stack collects task summary card, assumptions card, open issues, menus for generic EYA and Telerobotics primitives, syntax argument menus, and specific primitives cards for one task. Used to specify the task primitives for any task, and drive out technology requirements. Derivatives of that task (different assumptions, outline, tools, etc.) are quickly created, and the technology requirements analysis re-run.

## **JPL** TEJAS Statement of Purpose

---

• Create a clear, consistent, repetitive, interactive and exchangeable method of relating EVA and telerobotics, on various levels:

- Activity: Task, Subtask, Primitives
- Environment
  - On-orbit astrophysical medium
  - Worksites and taskboards
- External resources
  - Tools and services, jigs etc.
- Work Systems
  - FTS, JEMS-RMS, JEMS-SFA, MRMS, RMS, SPDM
- Technologies
  - Sensing/Perception
  - Planning/Reasoning
  - Control/Execution
  - Operator Interface
  - System Architecture

**FY 1989  
Emphasis**

## **JPL** TEJAS FY 1989 Scope

---

**Telerobotics  
Tasks**

**Telerobotics  
Assumptions**

**Telerobotics  
Open Issues**

**Telerobotics  
Primitives**

**Telerobotics  
Technologies**

**Telerobotics  
Systems**

**EVA  
Primitives**

**EVA  
Tasks**

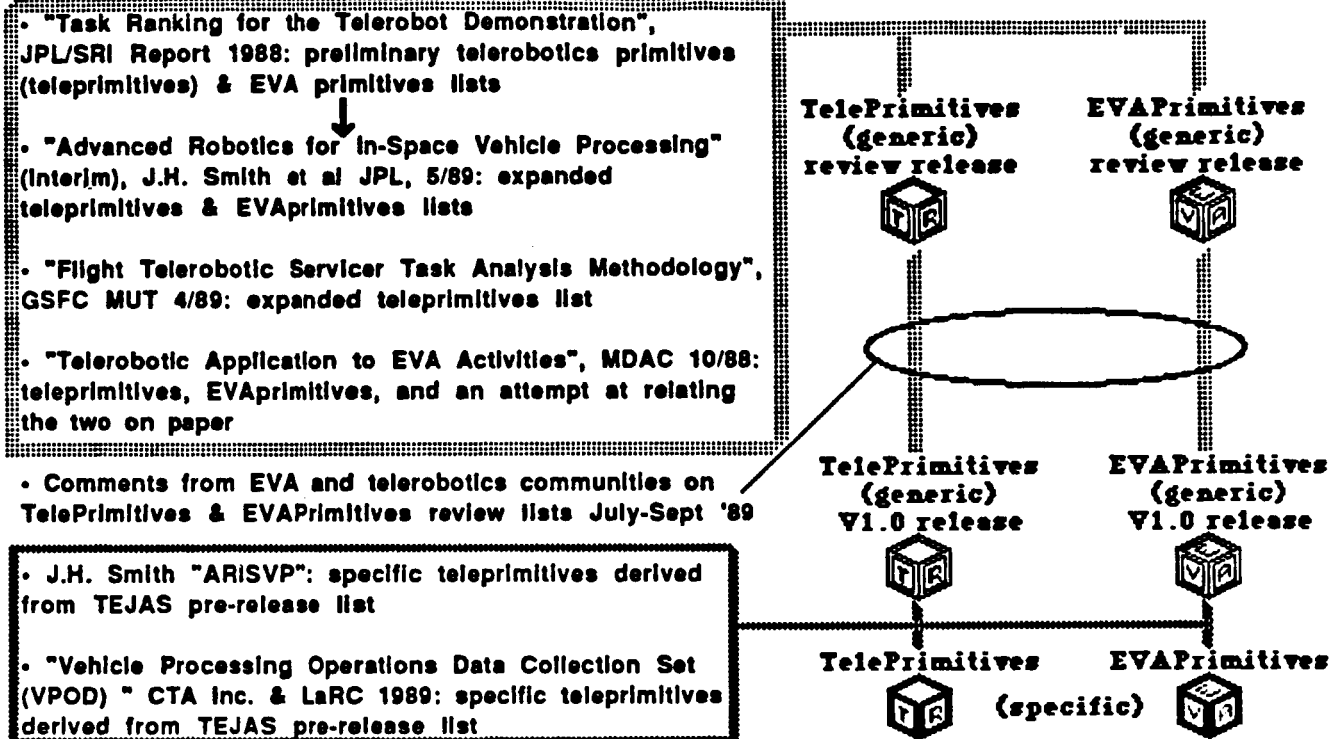
**EVA  
Assumptions**

**EVA  
Open Issues**

**Telerobotics  
& EVA  
Acronyms &  
Abbreviations**

**Telerobotics  
& EVA  
Glossary  
of Terms**

## JPL Generic Task Primitives' Evolution

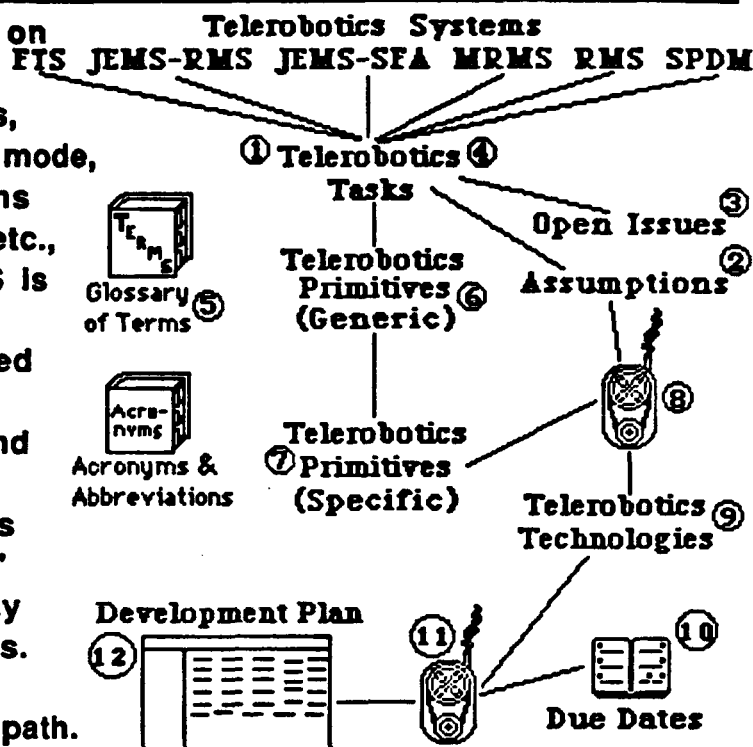


## JPL EVA and Telerobotics Primitives: Conclusions

- The basic steps of task analysis methodologies (TAMs) are similar between EVA and Telerobotics, up to a point (the outline).
- Primitives are universal objects of task analysis, with their *here/* set in each particular methodology. Primitives should be standardized, not the TAMs.
- Two kinds of primitives exist: *generic* and *specific*. Generic primitives have generic arguments. Specific primitives follow by specifying these arguments. Generic primitives are therefore a finite set of roots; specific primitives change according to each task, and are a virtually unlimited set.
- Verb-based syntaxes for standardizing and relating generic primitives must be part of any compilation. They help automate the generic-->specific step.
- EVA primitives are far more complex, but less numerous, than telerobotic ones, because a human can sense and identify, move etc. (multitasking).
- Links between EVA and telerobotics are through task- *specific* primitives.

## JPL Specifying Technology Requirements

**Technology requirements depend on task analysis. Specification of:** **FTS**  
**a) arguments of generic primitives, including work system, operating mode conditions, etc. and b) assumptions about the environment, the task, etc., will feed a decision matrix. TEJAS is RELATIONAL by design, not hierarchical. ANY hierarchy desired may be imposed on it without disturbing its built-in versatility and components. The numbered steps indicate that deriving technologies may involve relationally 'skipping' about the stacks as needed, in any order the thought process requires. The end result is a technology development plan, with a critical path.**



# JPL Specifying Technology Requirements

Technology requirements derive from a need to perform a task under certain circumstances. The TEJAS software approach depends on the hypermedia program HyperCard to create specific verb primitives from generic ones, thereby relating the task at that level with assumptions, external resources (e.g. information on tools, services, acronymns, terms etc.), the environment, and the telerobotic system(s) used (FTS, JEMS-RMS, JEMS-SFA, MRMS, RMS, SPDM). Relational rules in an analysis stack are then built to identify the required technologies. In determining the critical path of development, technology forecasts drive the telerobot calendar; the telerobot's selected missions impose the due dates.

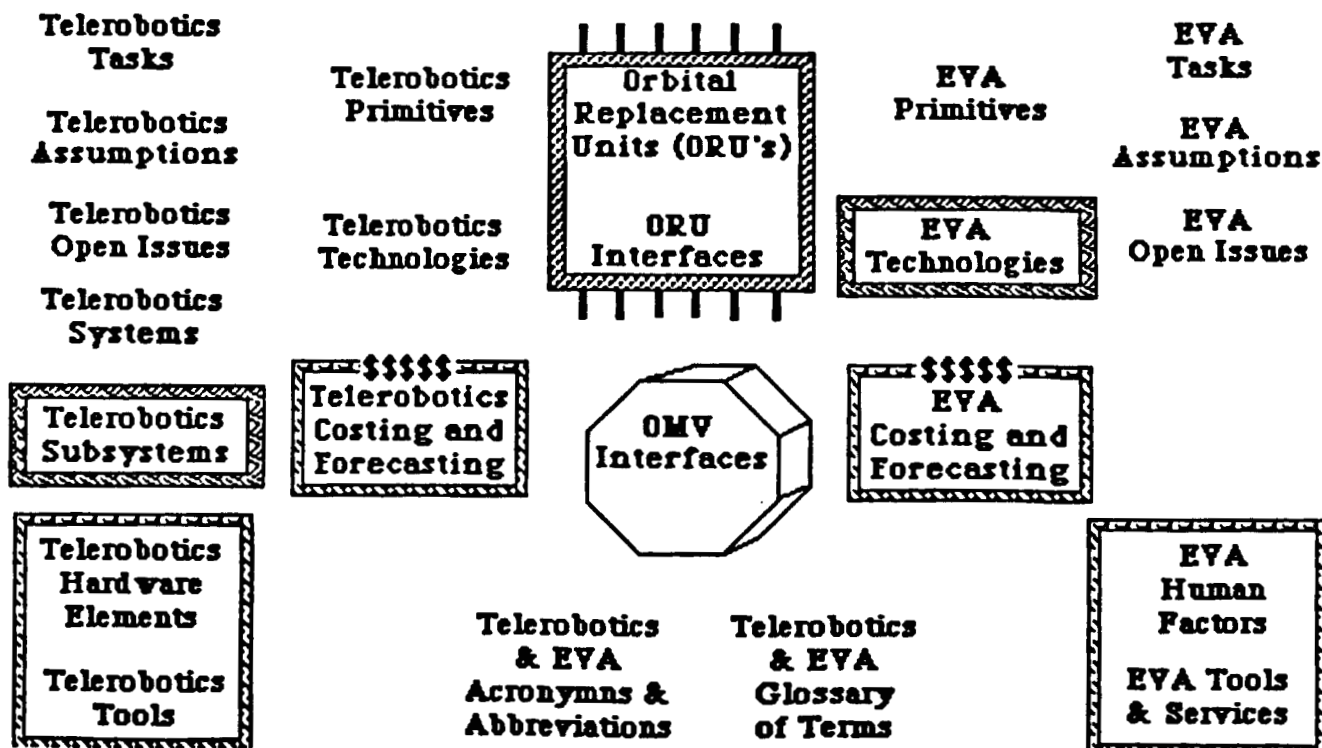
**To test its accuracy, FY89 TEJAS efforts focused on three real on-orbit future tasks: Maintenance, Servicing and Assembly (with a payload) of the Orbital Maneuvering Vehicle (OMV). Results will be published with our Version 1.0 final report.**

## JPL TEJAS Version 1.0 Limits

What TEJAS V1.0 will NOT do:

- Analyze Tasks with BOTH telerobots and EVA crew performing work. For this version, it's either/or. But FY90 efforts should make a mix possible, right down to specific primitives, while retaining the analysis capabilities as they stand. Telerobots and EVA crew can then be analyzed while interacting both serially AND in parallel. This is realism - the optimum. Telerobots will probably not fully supplement EVA crew for some time, but they are already working in concert with EVA (e.g. the RMS). This modification will allow realistic analysis of 'mixed' EVA/Telerobot tasks.
- Create task timelines, although specific primitives have fields in them for time, frequency (within a task), etc., and can hold many more parameters. But automated timeline-building simply wasn't part of our FY89 objectives. It easily can be part of FY90's.
- Cost forecasting for Telerobotics and/or EVA. Some of this is planned for FY90 as well. Its accuracy depends on realistic cost data - VERY scarce, and questionable when available. This may be the hardest goal of all.

## JPL TEJAS Proposed FY 90 Scope



## **JPL** TEJAS FY 90 Primary Objectives

---

- Use and mature the task analysis stacks, particularly the Assumptions stacks. Expand them to better define the environment and constraints, using some of the yet-to-be developed stacks (next page) as references.
- Further develop primitives stacks with feedback from the EVA and telerobotics community. Incorporate missing verbs, better syntaxes, etc.
- Further develop TeleTechnologies stack. Identify missing technologies, complete all forecasting (with histograms), and analyze. Build in rules for automated derivation of technology requirements for each task, with trace.
- Further develop TeleSystems stack. Provide summary comparison matrices between TeleSystems for physical characteristics and telerobotics technologies incorporated. Provide extensive relational links.
- Continue analysis of three OMV tasks, evolving better menus for specifying primitive arguments. Emphasize a realistic telerobotics/EVA parallel/serial operations mix.

## **JPL** TEJAS FY 90 Secondary Objectives

---

- Develop the following stacks:

### ExtraVehicular Activity:

- EVA Tools & Services - scanned stack copy of the popular catalog
- EVA Human Factors - scanned background data on reach, fatigue, etc.
- EVA Technologies - quantified stack for EVA, with forecasting (maybe)

### Telerobotics:

- Telerobotics Subsystems - breakdown for each work system
- Telerobotics Hardware Elements - joints, links, motors, sensors, etc.
- Telerobotics Tools - similar to the EVA Tools & Services Catalog
- Telerobotics Costing and Cost Forecasting - addendum to the TeleTechnologies stack, plus some background algorithms

### Spacecraft:

- Orbital Replacement Units - a reference stack, based on the results of the ongoing efforts in ORU standardization
- ORU Interfaces - scanned data on fasteners, plugs, surfaces, umbilicals, etc. Complimentary to both the EVA and Telerobotics Tools stacks.
- OMV Interface - scanned details on the interfaces referenced in the tasks

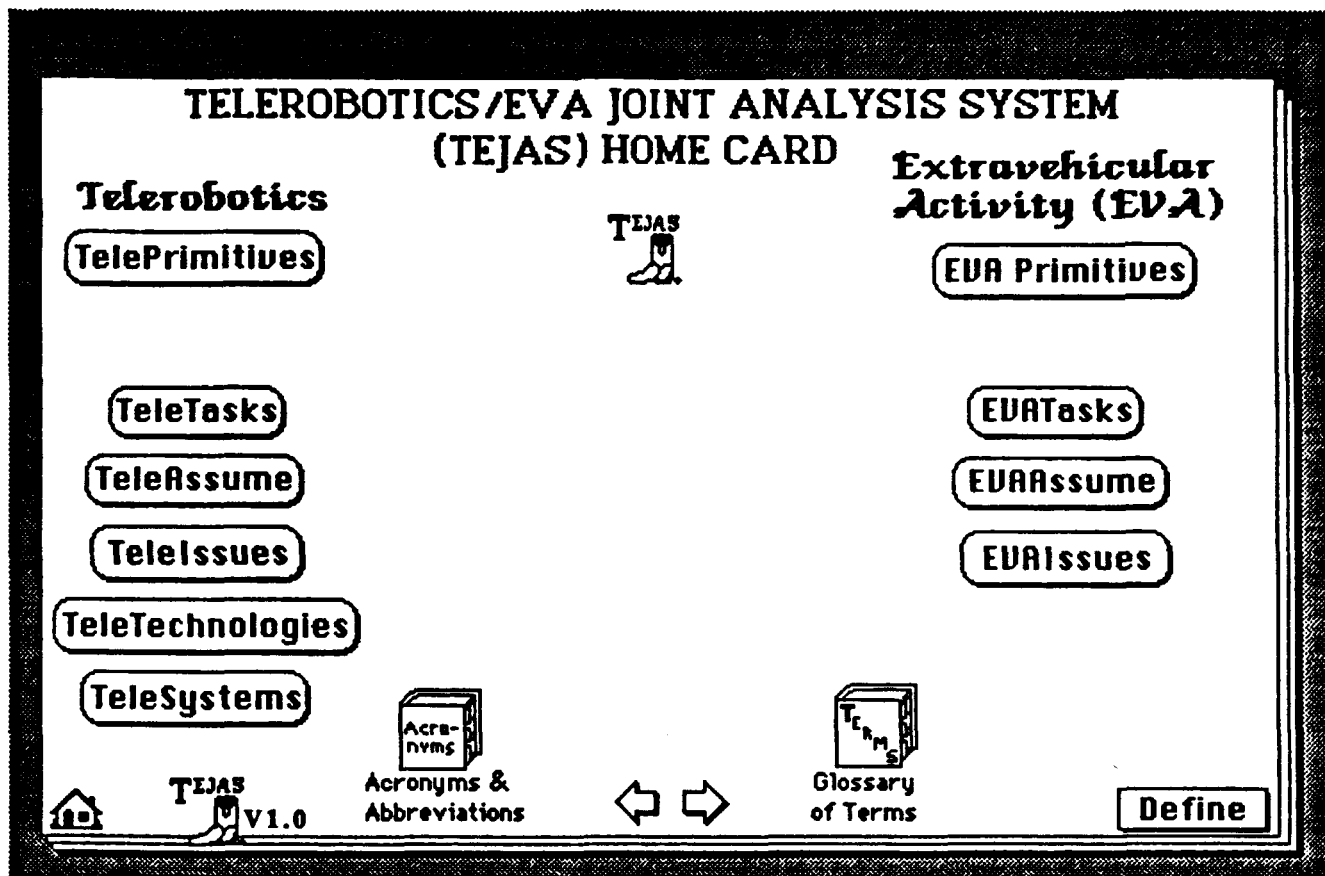
**JPL**

---

The TEJAS Stack System

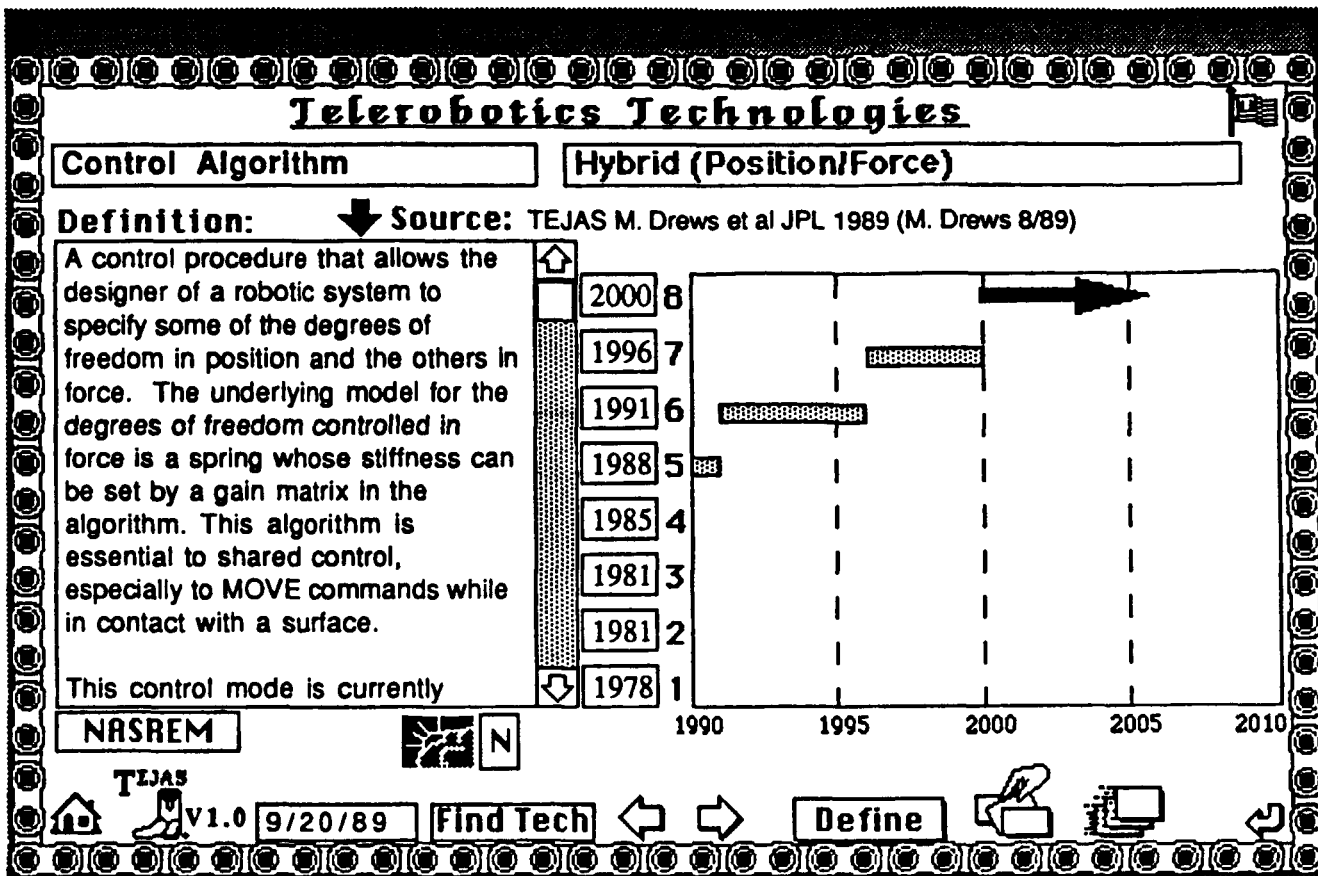
**JPL**

---



## JPL TEJASHome Stack

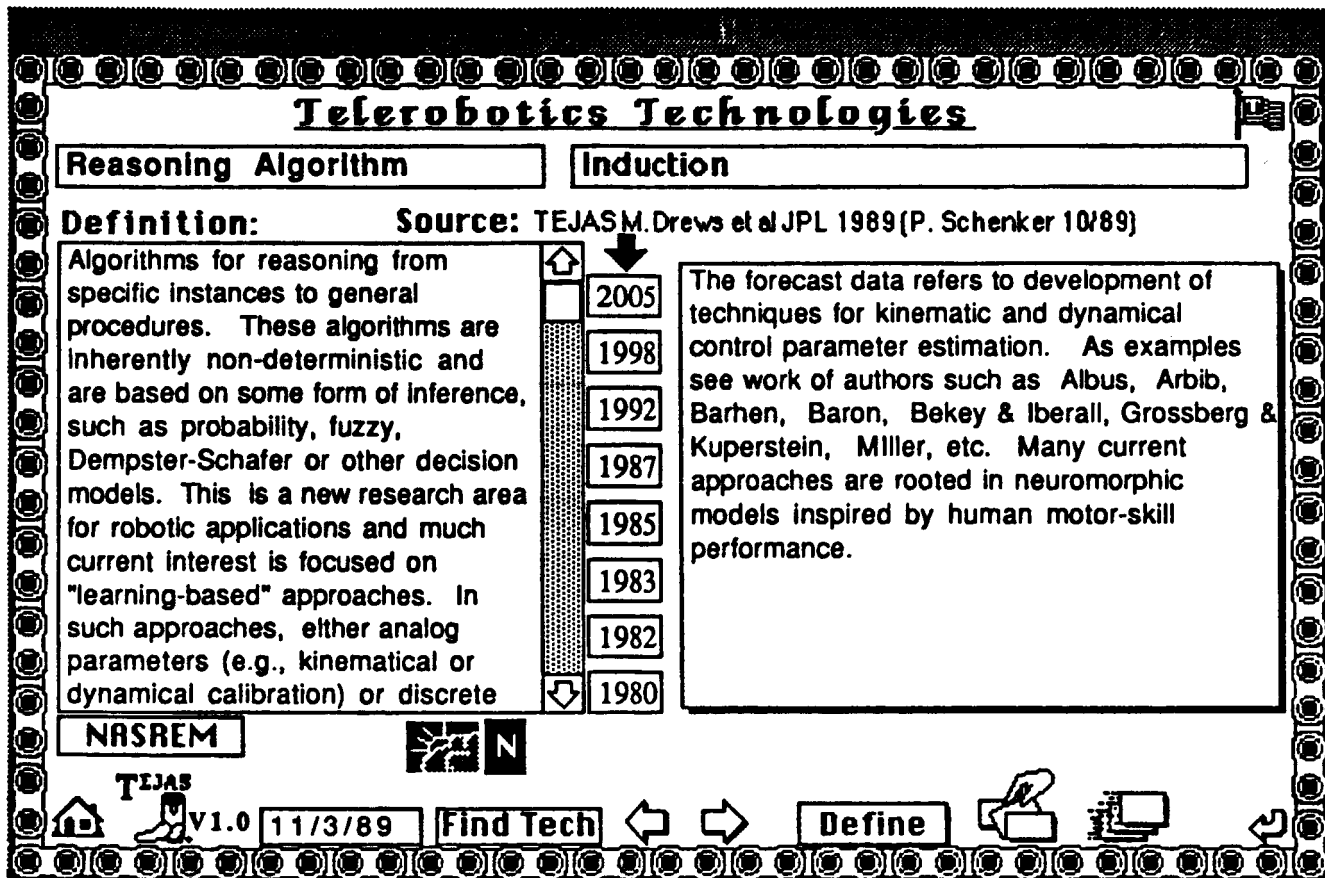
Every system needs a root directory, a bottom line, a home base. In TEJAS, this is it. Most of the time, a user will never need to go back here. But if ever they get lost in the stacks, pressing on the TEJAS boot in the lower left-hand corner of EVERY card in EVERY stack will get them there. If they continue to press the boot, a menu appears with the stacks as choices (including several choices of 'Help'). Pulling up and releasing on any stack takes you directly to the first card of that stack. It's another, simpler way to get around, bypassing the TEJASHome card and saving time.



## JPL TeleTechnologies Stack

This stack is an in-depth attempt at compilation and standardization of information about the technologies inherent in telerobotics. The cards store information about each technology, including a definition and a forecast of availability, with eight different levels of readiness, based on a system from "The Human Role In Space" (THURIS) study of 1982. The source field is actually three, superimposed and hidden until needed, to save space. They tell where we got the technology, its definition, and the forecast data. Another hidden field (Forecast Notes) puts restrictions on that forecast, if there are any.

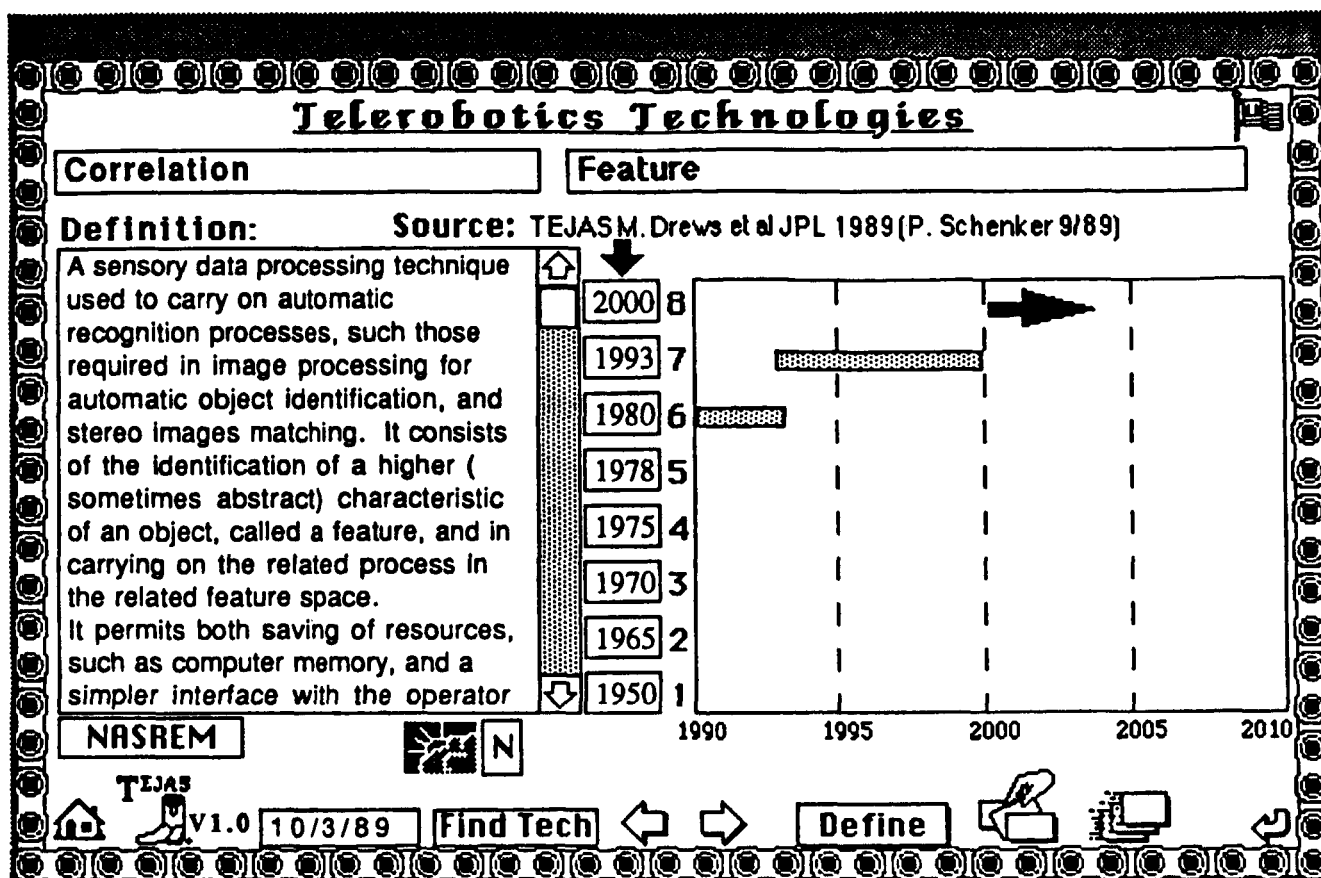
Although no standard of technology readiness levels has yet been specified by NASA, this system has been used extensively within Space Station studies. If another system is desired, it can easily be overlaid on the current information. The data shown can be changed, including the histograms, at any time. In FY90, we hope to better solidify this forecasting, and add approximate costs for each level.



## JPL TeleTechnologies Criticality

Previous technology studies have attempted to create lists of critical technologies, justifying the criticality (rank) of each on the circumstances of the study. But that leaves out a full treatment, because the "tall poles in the tent" are often more than the author has divined. Sometimes it is as important to know ALL of the technologies involved, critical or not, when assessing a situation.

TEJAS works differently to show the critical technologies. Once a task is broken down into verb primitives with the arguments specified, and the task assumptions stated, scripts take this data and, using a set of decision criteria (to be user-adjustable in Version 2.0, but hard set in Version 1.0), specify the requirement for each technology for each primitive. These are compiled into a single list for the task. Then, based on the task mission date, the scripts show all technologies that will NOT reach Level 8 (full operational capability) by that date, in order of how many years behind they'll be. Change your forecast, rerun the analysis, and you'll get a different ranking. It's that simple.



## **JPL** TeleTechnologies Stack - More to Come

This stack is meant to provide the reference point for many such analyses. There are HUNDREDS of technologies involved in telerobotics. Telerobot work systems implement different sets of these, depending on the designers and their criteria. On-orbit tasks require different sets. For any task, change the assumptions, change the conditions, change the telerobot, and the conclusion of provision versus requirements for telerobot technologies will change too.

We do not claim to have covered ALL of the technologies involved in telerobotics, in this version of the stack. But those that are missing can be ADDED, quickly and easily, and take their place in the analysis. Treating technologies as objects gives you this flexibility.

FY90 improvements may include decision matrices for technologies, because they are interrelated. They can be competing, mutually exclusive, or mutually reinforcing. These relationships are the discriminators needed to decide what should be pursued, and when.

## Telerobotics Technologies

Reference Frame

Object geometric center

**Definition:** Algorithms for resolution of manipulator end effector motion to a cartesian, polar, or cylindrical geometric reference grid centered at and oriented to the geometric (volumetric) center of an object being manipulated, normally to facilitate rotational motion about this point, so as to improve geometric understanding of the motion by the operator. Implies knowledge of the location of the object's geometric center, within some tolerance. Normally used to

**Source:** TEJAS M. Drews et al JPL 1989 (P. Schenker & M. Drews 10/89)

1982 8  
 1981 7  
 1975 6  
 1972 5  
 1970 4  
 1965 3  
 1960 2  
 1950 1

Currently Available  
Level 8

1990
1995
2000
2005
2010

NASREM

N

TEJAS  
V1.0

11/3/89

Find Tech

Define

## TeleTechnologies Contributions

**NOTES TO THE READER:** If you contribute a technology or definition to this stack and list its source, that's great. Thank you. But if you contribute ALL of the technologies or forecast data from a source, or cross-correlate that source, that's even better, and needs to be noted here, so we know at a glance what this stack includes. Enter the source name & date with appropriate notes on a numbered line at left below, and your name & date of entry on the corresponding line, please.

COMPLETE SOURCES CHECKED/COMPILED	CONTRIBUTOR & DATE
1. "Flight Telerobotic Servicer Task Analysis Methodology", GSFC Mission Utilization Team, 4/14/89;	1. M. Drews Jul 89
2. "Space Station Automation Study - Satellite Servicing" Vols. I & II, TRW, NAS 8-35081 (SN 41050) 11/84; Automated Servicing Technology Assessment used	2. M. Drews Jul 89
3. "Human Factors in Engineering and Design", E.J. McCormick & M. S. Sanders, McGraw-Hill, New York, 1982; used as specific reference	3. M. Drews Jul 89
4. "Space Station Automation", Proceedings of SPIE, Vol. 580, 9/85; scanned for technology evaluations	4. M. Drews Aug 89
	5. M. Drews Aug 89
	6. M. Drews Aug 89
	7. P. Fiorini Aug 89

TEJAS  
V1.0

9/24/89

ZEnd

EVA Primitives		INSTALL	
<b>Syntax</b>	<b>Source</b> D. Klaus JSC comts on "T/R & EVA Task Anal"		
(crew) INSTALL (object_A) in (object_B)			
<b>Definition, Notes etc.</b>			
To attach one object in or on another (i.e. a cover, a protective envelope such as a micro-meteor shield or MLI, fasteners, etc.), either by hands-on or with the use of a manual or powered tool. This definition is to cover other attachment activities that would otherwise require a large number		<b>ORIGINAL DEFINITION (from Source):</b> To fix in position for use.  <b>SUGGESTED ADDITION. KEEP AS IS, CHANGE, OR LEAVE OUT.</b>  MLD: probably needed; leave as is (	
<b>Like Verbs:</b>	<b>Antonym</b>		
MATE		REMOVE	
SECURE			
	<b>History</b>		
	<b>Find Verb</b>		
<div> </div>			

## JPL EVAPrimitives Stack

The EVAPrimitives stack serves as a repository for a final set of generic EVA primitives, with a definition, syntax, like (similar) verbs (NOT synonyms), an antonym, and other information that all can agree on. This is where the real power of relational hypermedia can be seen. Each card can be changed, with new or improved fields, or graphics, animation, and sound added if necessary, to completely convey the idea. When one is copied into any other stack, all that goes with it!

This stack is envisaged as a master list of GENERIC primitives, accessed during task analysis from the TeleTasks stack via a menu. SPECIFIC primitives evolve by specifying the arguments ACCORDING TO THAT PARTICULAR TASK, leaving the master list intact - a powerful idea. EVA primitives are much more complex than telerobotics primitives, because complex humans can deal with them more easily as basic building blocks of tasks than a telerobot can. Compare the syntax parsing schemes for each to see this.

EVA Primitives		COLLECT
Syntax	Source "Telerobotic Application to EVA" MDAC 10/88	
Definition, Notes etc.		
DELETED	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>	ORIGINAL DEFINITION (from Source): To gather objects together. (crew) COLLECT (objects)  JPL: Requires better definition; too ambiguous to be useful at this time. Comments, please.
Like Verbs:	Antonym	
	History	
	Find Verb	
TEJAS V1.0 <span>11/1/89</span> <span>Define</span>		

EVA Primitives Contributions Card	
<p><b>NOTES TO THE READER:</b> If you contribute a term and definition to this stack and list its source, that's great. Thank you. But if you contribute ALL of the terms in an existing document, or cross-correlate that source, that's even better, and needs to be noted here, so we know at a glance what this stack includes. Enter the source name &amp; date with appropriate notes on a numbered line at left below, and your name &amp; date of entry on the corresponding line, please.</p>	
COMPLETE SOURCES CHECKED/COMPILED	CONTRIBUTOR & DATE
1. "Telerobotic Application to EVA Activities", MDC H4121, McDonnell-Douglas Astronautics Co., 10/88; all terms/defs checked/compiled	1. J. Estus / C. Heneghan Jun 89
2. "Advanced Robotics for In-Space Vehicle Processing" (Interim), J.H. Smith et al JPL, 5/89	2. J. Estus / C. Heneghan Jun 89
3. "Telerobotics/EVA Joint Analysis System (TEJAS)" (Pre-release), M. Drews et al JPL, 6/89; used for cross-correlation & composition	3. M. Drews Jun 89
4. "Flight Telerobotic Servicer Task Analysis Methodology", GSFC Mission Utilization Team, 4/14/89; checked, no EVA primitives	4. C. Heneghan Jun 89
	5. J. Smith Jun 89
	6. J. Smith Jun 89
TEJAS V1.0 <span>11/1/89</span> <span>ZEnd</span>	

Telerobotics Primitives		GRAPPLE	
<b>Syntax</b>		<b>Source</b> FTS Task Analysis Methodology 4/14/89	
(subsystem) GRAPPLE (object_A)			
<b>Definition, Notes etc.</b>			
To close the snares on a cable-tension end effector, such as the Orbiter RMS, around a compatible (grapple) fixture or object, and then to retract the snares, in order to draw tight and snug the payload (to which the grapple fixture or object is attached) to that end effector (GRAPPLE =		<b>ORIGINAL DEFINITION (from Source):</b> To attach a sub-system to an object using a grapple fixture.  Antonym: RELEASE  JPL: replace 'sub-system' with 'enclosure'	
<b>Like Verbs:</b> CAPTURE RIGIDIZE GRASP	<b>Antonym</b> RELEASE		
	<b>History</b> <b>Find Verb</b>		
	NASREM	Operator Interface	Control Execution
<div> </div>			

## JPL TelePrimitives Stack

The TelePrimitives are similar to the EVA Primitives, i.e. generic in arguments. The definition, syntax, and other information must be much more exact, requiring more optional arguments, leading therefore to almost an infinite set of potential specific teleprimitives. Therefore, like the EVA primitives, these arguments are specified via a pull-down menu (with alternate entry possible) during the task analysis. For each task, a set of specific teleprimitives must be built in this manner, one by one, to construct an approximation of the task outline.

Obviously, the relationship between EVA and Telerobotics primitives is a one-to-many. We have attempted to build that link in the direction EVA->Telerobotic. On the generic level, it is difficult, but our attempts show promise. The significance is that should a hard relationship set be attainable, we'll be able to use it to trade off using EVA or telerobots in a task. Much more needs to be done here, in FY90.

FY 90 Plan

POP 89-2 (FY 1990)												DATE: 1 NOV 89	
CODE ST: ADVANCED DEVELOPMENT TASK DESCRIPTION												JPL ACCOUNT CODE: 293-90001-0-3470	
TITLE: Telerobotics/EVA Joint Analysis System												TASK MANAGER: M. Drews	
UPN: 476-14												TECHNICAL MANAGER: B. HANSEN III	
<b>TASK DESCRIPTION:</b> The objective of this task is to continue development of a realistic technology development planning tool using hypermedia, expanding the analysis beyond the first three FY89 on-orbit servicing tasks, incorporating more data (especially technology forecasts), more relational rules, and costing analysis. This will allow comparison between alternative technology development plans to find one that can fit real-world budget and program constraints. Timelines, cost projections, matrix summaries, and recommendations for technology development will be supported, for any number of tasks.													
<b>TECHNICAL APPROACH:</b> This project uses hypermedia stacks to create detailed references of EVA and telerobotics task Primitives* and tools and equipment, telerobotics Technologies* (each with a forecast), Systems*, and hardware elements, EVA and telerobotics costs and cost forecasting, and Orbital Replacement Unit (ORU) interfaces. Analysis stacks, such as the EVA and telerobotics Tasks*, Assumptions*, and Open Issues*, allow the user to interactively build a specific set of verb primitives for any particular task, relate those to each other, derive the technologies required, and then identify the critical path of technology development by comparing each required technology's forecast against the mission date for that task. The FY90 effort continues analysis and development of all current stacks (* above), creates the other reference stacks and builds links to other applications.													
NOA	PRIOR	FY89	FY90	FY91	FY92	FY93	FY94	FY95	FY96	FY97	BTC	TOTAL	
		150	150									300	
PRIOR OBLS			75									75	
CY OBLS			75	150								225	
COSTS			75	225								300	
JPL M/P			0.5	1.8								2.3	
CTR M/P													
K \$													
FY 90	OCT	NOV	DEC	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	
NOA	150											150	
PRIOR OBLS (CUM)	18.5	37.5	56	75								75	
CY OBLS (CUM)					18.5	37.5	56	75	93.5	112.5	131	150	
COSTS (CUM)	18.5	37.5	56	75	93.5	112.5	131	150	168.5	187.5	206	225	
JPL M/P	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	
CTR M/P													
NOTES:													

**Title:** Telerobotics/EVA Joint Analysis System (TEJAS)  
**UPN:** 476-14

### **Technical Approach:**

Telerobotics technology requirements derive from a need to perform an on-orbit task under certain circumstances. A technology development plan must take those required technologies and the need date, compare them against real-world constraints (such as the technology's readiness level, and the way the telerobotic system used for that task does or does not incorporate it), and prioritize the expenditure of time and money to make the readiness level date for Level 8 (on-orbit availability) or Level 7 (on-orbit test) meet the need date. Before any of this can be done, an extensive database of technologies, telerobotic work systems, task primitives, tools, etc. must be constructed.

The TEJAS software approach depends on the hypermedia program HyperCard to specify and link a task (through its specific task primitives) with the assumptions, peripheral information (e.g. tools, services, acronyms, terms etc.), and telerobotic work system(s) used (FTS, JEM-RMS, JEM-SFA, RMS, SSRMS, SPDM). Relational rules are then built into those links (via an analysis stack) to help the user identify the required telerobotics technologies.

Getting to the *specific* primitives requires building hypermedia files (called 'stacks') that compile, define, standardize (through use of a common verb-oriented syntax), and interrelate existing *generic* Telerobotics and EVA verb primitives from the major task analysis activities nationwide. The first version of these TelePrimitives and EVAPrimitives stacks was one of the major deliverables from the 1989 TEJAS effort. Also created in that effort, the TeleTechnologies stack identifies, defines, and forecasts over 200 different technologies inherent to telerobotics. An EVA and telerobotics Glossary of Terms (over 1100), and Acronyms & Abbreviations (over 900) dictionary are well on their way to becoming NASA standard documents.

**Title:** Telerobotics/EVA Joint Analysis System (TEJAS)  
**UPN:** 476-14

**Technical Approach (cont'd):**

All of these reference stacks must now be released to and reviewed by the telerobotics and EVA community. Many changes (especially additions) will result. The work of inputting that data and making new capabilities available in the stacks (for example, keyword searches) will be a significant part of the FY90 effort. Among the expected changes are new primitives, new technologies, and updated readiness forecasts for those technologies (many could not be finished in FY89). The FY90 TEJAS activity will also create additional HyperCard stacks and input the information in the areas of: Telerobotics Tools, End Effectors, Telerobotics Subsystems (for each work system), Telerobotics Hardware Elements, EVA and Telerobotics Costing and Cost Forecasting. These databases will then be related to the existing TEJAS tool and to each other. For each of the three OMV servicing test cases (or other tasks if specified), the tool will be used to completely define the task and all parameters, then regenerate a telerobotics technology development plan. Costing analysis using the integrated cost forecasting data will show cost curves by Fiscal Year for developing the technologies necessary to do the task with the assumptions detailed.

Call No:

TL

797

.349

. 1990

v.2

pt.2

c.1

Title: Telerobotics/EVA Joint Analysis System (TEJAS)  
UPN: 476-14

**Inputs:**

	<b>From:</b>	<b>Date:</b>
1989 TEJAS Version 1.0 stacks etc.	JPL Section 347	11/1/89
EVA Costs information	NASA JSC & contractors	1/1/90
Telerobotics Technologies review & forecasts	NASA GSFC, JSC, ARC, LaRC	2/1/89
End Effector Specifications	NASA, NASDA, CSA	2/1/90
Telerobotics Subsystems information (by Work System)	NASA, NASDA, CSA	2/1/90
Telerobotics Technologies Costs information	NASA GSFC, JPL, ARC, LaRC	4/1/90

**Products/Deliverables:**

	<b>To:</b>	
On-orbit Task Analysis by EVA and Telerobotics	NASA OSS (Code ST)	4/30/90
Telerobotics Technologies Critical Development Path Plan	NASA OSS (Code ST)	4/30/90
Mid-Term Report (TEJAS Version 1.5 mid-term release)	NASA OSS (Code ST)	4/30/90
EVA and Telerobotics Costs & Cost Forecasting	NASA OSS (Code ST)	9/30/90
1990 TEJAS Version 2.0 stacks, relations, etc.	NASA OSS (Code ST)	9/30/90
Final Report	NASA OSS (Code ST)	9/30/90
TEJAS User's Manual	NASA OSS (Code ST)	9/30/90

Detailed schedule attached.

# WORK PACKAGE AGREEMENT SCHEDULE/PLAN SUPPORTING DETAIL

ITEM	ACTIVITY/MILESTONE	ORG	O N D J F M A M J J A S												ACCOUNT CODE	VERSION	DATE	PAGE
			O	N	D	J	F	M	A	M	J	J	A	S				
1	TEJAS STACK DEVELOPMENT																	
2	TeleSystems Subsystems																	
3	Telerobotics Technologies Review																	
4	& Forecasts																	
5	End Effector Specifications																	
6	Telerobotics Tools Catalog																	
7	On-Orbit Task Analysis																	
8	EVA Costing/Cost Forecasting																	
9	Telerobotics Costing/Cost Forecasting																	
10																		
11	Technology Development Timelines																	
12	Technology Development Cost Analysis																	
13	Technology Tradeoffs Analysis																	
14	Mid-Term Report & TEJAS V1.5 Release																	
15	Final Report & TEJAS V2.0 Release																	
16	TEJAS Version 2.0 User's Manual																	
17																		
18																		
19																		
20																		
21																		
22																		
23																		
24																		
WORK PACKAGE TITLE																	FY90	
Telerobotics/EVA Joint Analysis System (TEJAS)																	7/13/89	
																	5 OF 5	

# REPORT DOCUMENTATION PAGE

1. Report No. NASA CP-10044		2. Government Accession No.		3. Recipient's Catalog No.	
4. Title and Subtitle Beyond the Baseline - Proceedings of the Space Station Freedom Evolution Symposium at League City, Texas, February 6, 7, & 8, 1990 Volume 2: Space Station Freedom Adv. Dev. Prog.; Part 2				5. Report Date May 1990	
				6. Performing Organization Code HQ MT	
7. Author(s) NASA and support contractor personnel funded under the Transition Definition line item - UPN 488				8. Performing Organization Report No. S-606	
9. Performing Organization Name and Address Space Station Engineering NASA Headquarters Washington D.C. 20546 MC:MT				10. Work Unit No.	
				11. Contract or Grant No.	
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington D.C. 20546				13. Type of Report and Period Covered Conference publication	
				14. Sponsoring Agency Code	
15. Supplementary Notes					
16. Abstract This report contains the individual presentations delivered at the Space Station Evolution Symposium in League City, Texas on February 6, 7, 8, 1990. Personnel responsible for Advanced Systems Studies and Advanced Development within the Space Station Freedom Program reported on the results of their work to date. Systems Studies presentations focused on identifying the baseline design provisions (hooks and scars) necessary to enable evolution of the facility to support changing space policy and anticipated user needs. Also emphasized were evolution configuration and operations concepts including on-orbit processing of space transfer vehicles. Advanced Development task managers discussed transitioning advanced technologies to the baseline program, including those near-term technologies which will enhance the safety and productivity of the crew and the reliability of station systems. Special emphasis was placed on applying advanced automation technology to ground and flight systems.					
17. Key Words (Suggested by Author(s)) evolution; "hooks and scars"; advanced automation; knowledge-based systems; robotics				18. Distribution Statement Unclassified - unlimited  Subject category: 18	
19. Security Classification (of this report)  U		20. Security Classification (of this page)  U		21. No. of pages  445	
22. Price					

For sale by the National Technical Information Service Springfield, VA 22161 2171